НАЦІОНАЛЬНИЙ ТЕХНІЧНИЙ УНІВЕРСИТЕТ УКРАЇНИ
«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

На правах рукопису
УДК 519.688

До захисту допущено
В. о. завідувача кафедри ММСА
О.Л.Тимощук
«____» _____ 2019 р.

Магістерська дисертація

на здобуття ступеня магістра за спеціальністю 124 Системний аналіз
на тему: «Методи управління в когнітивних картах
на основі прогнозуючих моделей»

Виконав: студент ІІ курсу, групи КА-81мп
Міщенко Михайло Дмитрович _____

Керівник: професор кафедри ММСА,
д.т.н., проф. Губарев Вячеслав Федорович _____

Рецензент: старший науковий співробітник
ІКД НАН та ДКА України,
к.т.н. Микола Миколайович Сальников _____

Засвідчую, що у цій магістерській
дисертації немає запозичень з праць
інших авторів без відповідних посилань
Студент _____

Київ
2019

«КИЇВСЬКИЙ ПОЛІТЕХНІЧНИЙ ІНСТИТУТ ІМЕНІ ІГОРЯ СІКОРСЬКОГО»
ІНСТИТУТ ПРИКЛАДНОГО СИСТЕМНОГО АНАЛІЗУ
КАФЕДРА МАТЕМАТИЧНИХ МЕТОДІВ СИСТЕМНОГО АНАЛІЗУ

Рівень вищої освіти —другий (магістерський)
Спеціальність —124 «Системний аналіз»

ЗАТВЕРДЖУЮ
В. о. завідувача кафедри ММСА
О.Л.Тимощук
«____» _____ 2019 р.

ЗАВДАННЯ

на магістерську дисертацію студенту Міщенкові Михайлу Дмитровичу

1. Тема дисертації: «Методи управління в когнітивних картах на основі прогнозуючих моделей», науковий керівник дисертації Губарев Вячеслав Федорович, д.т.н., професор, затверджені наказом по університету від «08» листопада 2019 р. № 3862-с.

2. Термін подання студентом дисертації: 13 грудня 2019 р.

3. Об'єкт дослідження: керовані лінійні системи та когнітивні карти

4. Предмет дослідження: методи синтезу керування для керованих лінійних систем та когнітивних карт на основі прогнозуючої моделі

5. Перелік завдань, які потрібно розробити:

1) оформити розглядаємі математичні моделі та постановку задачі керування у формальному вигляді;

2) оформити наявні теоретичні напрацювання у формальному вигляді;

3) провести дослідження щодо покращення розроблених методів синтезу керування;

4) провести дослідження щодо практичного застосування розроблених методів для керування системами;

5) провести чисельне дослідження роботи розроблених алгоритмів на модельних системах;

6) розробити концептуальні висновки за результатами наукового дослідження.

6. Орієнтовний перелік графічного (ілюстративного) матеріалу:

1) Графи процесу пошуку для ітераційного алгоритму пошуку розбиття за правилом послідовного обходу (рис.);

2) Графіки стану модельних систем при керуванні ними (рис.).

7. Орієнтовний перелік публікацій:

Публікація наукової статті у фаховому журналі "Кибернетика и вычислительная техника".

8. Дата видачі завдання: 05 вересня 2019 р.

Календарний план

№ з/п	Назва етапів вконання магістерської дисертації	Термін виконання етапів магістерської дисертації
1.	Концептуальний вступ дисертації. Формулювання об’єкта, предмета, цілі, завдань, новизни, практичної значущості результатів	18.03.2019—20.03.2019
2.	Перший розділ. Формалізація розглядаємих математичних моделей.	21.03.2019—30.03.2019
3.	Другий розділ. Постановка задачі керування.	31.03.2019—16.04.2019
4.	Третій розділ. Дослідження щодо практичного застосування розроблених методів для керування системами.	17.04.2019—06.05.2019
5.	Концептуальні висновки. Перспективи розвитку отриманих рішень.	07.05.2019—10.05.2019

Студент М.Д.Міщенко

Науковий керівник дисертації В.Ф.Губарев

РЕФЕРАТ

Магістерська дисертація: 87 ст., 1 табл., 13 рис., 14 джерел, 1 додаток.

ВАРІАЦІЙНЕ ЧИСЛЕННЯ, ДИСКРЕТНИЙ ЧАС, КЕРОВАНА СИСТЕМА, КЕРУВАННЯ ЗА ПРОГНОЗУЮЧОЮ МОДЕЛЛЮ, КОГНІТИВНА КАРТА, ЛІНІЙНА СИСТЕМА, СИНТЕЗ КЕРУВАННЯ, СКІНЧЕННИЙ ГОРИЗОНТ.

Об'єктом дослідження цієї роботи є керовані лінійні системи та когнітивні карти.

Метою написання роботи була розробка алгоритмів синтезу керування на основі ідей та підходів до керування за прогнозуючою моделлю.

Основною частиною дослідження була теоретична розробка методів синтезу керування шляхом формулювання та доведення відповідних теорем. Також була проведена перевірка роботи розроблених методів на модельних прикладах шляхом обчислювальних експериментів.

Було отримано новий клас методів, здатних приводити стан лінійної системи до нуля (або у випадку наявності збурень - до його околу) та стабілізувати функціонування когнітивної карти за скінченний час. Отримані методи здатні керувати не тільки строго стійкими системами, а також і напівстійкими та нестійкими, у тому числі і в умовах наявності випадкових збурень та з урахуванням обмеженості ресурсу керування.

Розроблені методи різняться за ефективністю використання ресурсу керування та необхідними обчислювальними ресурсами. Більш ефективні методи потребують більше обчислень для отримання результату. Це призводить до необхідності обирати оптимальний метод у кожному конкретному випадку окремо.

Ці методи можуть застосовуватися для керування як технічними, так і будь-якими іншими системами, що описуються як лінійні керовані системи з багатьма входами та виходами або як керовані когнітивні карти.

Розроблені методи залишають простір для подальших досліджень та покращень як самих обчислювальних алгоритмів, так і схем їх застосування.

ABSTRACT

Thesis: 87 p., 1 table, 13 images, 14 sources, 1 appendix.

CALCULUS OF VARIATIONS, COGNITIVE MAP, CONTROL SYNTHESIS, CONTROLLABLE SYSTEM, DISCRETE TIME, FINITE HORIZON, LINEAR SYSTEM, MODEL PREDICTIVE CONTROL.

Object of this thesis is controllable linear systems and cognitive maps.

The goal of this work is development of control synthesis algorithms inspired by ideas and approaches of model predictive control.

The main part of the research was formal designing of the control methods by formulating and prooving of corresponding theorems. Developed methods' testing with computational experiments on model systems was also performed.

Class of new methods capable of bringing linear system's state to zero (or in case of perturbation's presence - to its neighbourhood) or cognitive map stabilization in finite time was created. The developed methods are capable of controlling not only stable systems, but also metastable and unstable ones, even in presence of perturbations and while taking into account limitations on control resource, which are almost always present in real (not model) systems.

The developed methods vary in effectiveness of control resource utilization and required computational resources. More effective in this sense methods requires more computations to obtain a result. Because of this there is a necessity to choose an optimal method in every particular case.

This methods can be used to control both technical and any other kinds of systems represented either as controllable linear systems with multiple inputs and outputs or as controllable cognitive maps.

The developed methods leaves a field of futher research and improvement of both computational algorithms and schemes of their application.

ЗМІСТ

ПЕРЕЛІК УМОВНИХ ПОЗНАЧЕНЬ ТА МАТЕМАТИЧНИХ СИМВОЛІВ

$\overline{1..n}$ - множина цілих чисел від 1 до n. Замість n може стояти будь-яке ціле число або вираз, що дорівнює цілому числу.

card - потужність множини (кількість елементів у множині). Наприклад, запис card I_0 позначає потужність множини I_0.

dim - розмірність вектора чи квадратної матриці.

$\rightarrow$ - символ імплікації. Позначає бінарний логічний оператор, істинний тоді і тільки тоді, коли або лівий аргумент є хибним, або правий є істинним. Цей оператор є формальним описом поняття наслідку одного твердження із іншого.

const - деяка константа.

Im - образ оператора, множина усіх можливих результатів дії оператора.

Ker - ядро оператора, множина усіх величин, які дії даного оператора на котрі повертає $\mathbf{0}$.

$\langle \boldsymbol{a}, \boldsymbol{b} \rangle$ - скалярний добуток векторів $\boldsymbol{a}$ і $\boldsymbol{b}$.

Верхньою рискою над виразом, що позначає множину, позначається її замикання. Наприклад, вираз $\overline{U(I_0, I_{\min}, I_{\max})}$ позначає замикання множини $U(I_0, I_{\min}, I_{\max})$.

Посилання на деякі математичні формули, що в тому чи іншому вигляді задають множину, можуть використовуватися у інших формулах у якості умовного позначення цієї множини.

Запис A_I, де A - деяка матриця, а I - множина індексів, позначає матрицю, що складається з рядків матриці A, які відповідають індексам із множини I.

Запис $\boldsymbol{a}_I$, де $\boldsymbol{a}$ - деякий вектор, а I - множина індексів, позначає вектор, що складається з величин вектора $\boldsymbol{a}$, які відповідають індексам із множини I.

Запис A^I, де A - деяка матриця, а I - множина індексів, позначає матрицю, що складається зі стовпчиків матриці A, які відповідають індексам із множини I.

ВСТУП

Когнітивні карти є зручним засобом для моделювання динаміки систем. На даний момент було розроблено значну кількість методів для роботи з ними. Існують як методи керування такими моделями, так і методи ідентифікації їх коефіцієнтів та структури. Класичним підходом до керування ними є застосування методів теорії керування.

Основна ідея існуючих методів стабілізації когнітивних карт на основі теорії керування є утворення із системою замкненого контуру зі стійкою передавальною функцією. При цьому перед архітектором такого контуру постає низка питань без чіткої та однозначної відповіді. Одним із таких питань є те, яку швидкодію повинен мати цей контур, адже якщо вона буде надто високою, то виникатиме явище перерегуляції, коли надто активна реакція контролера на збурення може призвести до різкого стрибка стану системи в протилежному напрямку і як наслідок її руйнування. Другим відкритим питанням є те, яким має бути процес стабілізації системи: гладким чи коливальним.

Загальною ж проблемою цього підходу є те, що ці методи керування або детерміновані, або виходять із припущення про певні зручні статистичні властивості збурень. Якщо ж реальні збурення, що діють на систему, не мають таких властивостей, то стабілізувати систему за допомогою цих методів зазвичай не вдається.

Гідною альтернативою є застосування методології керування на основі прогнозуючої моделі. Основною її ідеєю є вибір такого керування, яке дасть найкращий прогнозований результат у майбутньому. Таким чином можна було б позбутися проблем, притаманних класичним методам стабілізації, адже у кожний момент часу обиралося найкраще за певним критерієм керування.

Розробці та дослідженню роботи таких методів присвячена дана робота. Отримані результати на даному етапі є корисними для стабілізації роботи систем, що описуються у вигляді когнітивної карти і мають повністю спостережний вектор стану.

Дана робота відкриває широкий простір для подальших досліджень, покращення та модифікації алгоритмів.

1 РОЗГЛЯДАЄМІ МОДЕЛІ

У цьому розділі детально описуються моделі систем, що розглядаються, а також пов'язані з ними задачі.

1.1 Когнітивна карта

Когнітивна карта задається міченим направленим графом, у якому допускаються петлі і в якому усі ребра помічені дійсними числами. Це - динамічна модель, що розвивається в дискретному часі. Вершинам графу відповідають коефіцієнти вектора $\boldsymbol{x}_k$, який задає стан системи в момент k. Кажуть, що кожній вершині по ребрам графу передаються імпульси інших вершин з відповідним коефіцієнтом. Стан системи у наступний момент часу $\boldsymbol{x}_{k+1}$ задається за формулою

$$\boldsymbol{x}_{k+1} - \boldsymbol{x}_k = A(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) + \boldsymbol{d}_k, \tag{1.1}$$

де A - матриця, що складається з коефіцієнтів відповідних ребер графу когнітивної карти, а $\boldsymbol{d}_k$ - скінченне випадкове збурення в момент k. Ті коефіцієнти A, у яких немає відповідних ребер, дорівнюють нулю. У детермінованому випадку вважаємо, що у кожен момент часу k вектор $\boldsymbol{d}_k$ рівний $\mathbf{0}$. Подальше функціонування системи також задається початковими умовами: значеннями векторів $\boldsymbol{x}_0$ та $\boldsymbol{x}_1$.

Когнітивна карта у постановці (1.1) функціонує незалежно від зовнішнього впливу. Проте, якщо додати можливість створення додаткових керуючих імпульсів $\boldsymbol{u}_k$, як у формулі

$$\boldsymbol{x}_{k+1} - \boldsymbol{x}_k = A(\boldsymbol{x}_k - \boldsymbol{x}_{k-1}) + B\boldsymbol{u}_k + \boldsymbol{d}_k, \tag{1.2}$$

постає цілий клас задач синтезу керування, про які піде мова у наступному розділі.

1.2 Лінійна система в дискретному часі з багатьма входами та виходами

Нехай є контрольована лінійна система в дискретному часі з багатьма входами та виходами. Вона може бути детерміністичною або стохастичною. Система задається формулою

$$\boldsymbol{x}_{k+1} = A\boldsymbol{x}_k + B\boldsymbol{u}_k + \boldsymbol{d}_k, \tag{1.3}$$

де $\boldsymbol{x}_k$ і $\boldsymbol{x}_{k+1}$ - послідовні у часі стани системи, а $\boldsymbol{u}_k$ - керування, застосоване до системи у k-й момент часу, $\boldsymbol{d}_k$ - випадкове збурення у момент часу k, A і B - матриці. Далі ми користуємось припущенням, що A і B мають повні ранги. Стани системи і керування - це n-вимірні та r-вимірні вектори відповідно. У детерміністичному випадку $\boldsymbol{d}_k$ завжди рівний $\mathbf{0}$.

1.3 Спорідненість моделей систем

Можна помітити, що якщо у (1.2) зробити заміну $\Delta\boldsymbol{x}_k = \boldsymbol{x}_k - \boldsymbol{x}_{k-1}$, то виходить формула

$$\Delta\boldsymbol{x}_{k+1} = A\Delta\boldsymbol{x}_k + B\boldsymbol{u}_k + \boldsymbol{d}_k, \tag{1.4}$$

яка повторює формулу для лінійної системи (1.3).

Таким чином, методи керування лінійними системами у ряді випадків можна застосовувати також до когнітивних карт.

1.4 Обмеження на керування

Якщо ми працюємо із реальною системою (а не з теоретичною моделлю), тоді значення координат векторів керування не можуть бути необмежено великими. Тому, необхідно задати певні обмеження на них. Ми задаємо обмеження згідно формули

$$|u_{ki}| \leqslant u_{\max i}, \quad k \in \mathbb{Z}, \quad i \in \overline{1..r}, \tag{1.5}$$

де $\boldsymbol{u}_{\max}$ - це вектор, що складається з додатніх значень.

Також вважаємо, що значення $\boldsymbol{d}$ обмежені таким чином, що вони не перевищують ресурсу керування, заданого у (1.5).

Ми задаємо обмеження в такому вигляді для усіх моделей систем, що розглядаються у цій роботі.

1.5 Висновки

У даному розділі було подано у формальному вигляді класи систем, про які йде мова у цій роботі, а саме: лінійні керовані системи у дискретному часі та керовані когнітивні карти. При формалізації було враховано принципову властивість реальних (не модельних) систем: обмеженість ресурсу керування.

2 ЗАДАЧІ, ЩО РОЗГЛЯДАЮТЬСЯ, ТА ПІДХОДИ ДО ЇХ РОЗВ'ЯЗАННЯ

2.1 Задачі для лінійної системи в дискретному часі з багатьма входами та виходами

Розглядається детерміністична або стохастична система, задана формулою (1.3). Відомий стан системи у момент k - $\boldsymbol{x}_k$. Перша задача полягає у тому, щоб привести стан системи у точку $\mathbf{0}$ (або, у стохастичному випадку, в її окіл) за скінченний час. Друга задача полягає в тому, щоб утримувати її в околі цієї точки необмежено довго за допомогою синтезованих керувань.

2.2 Задачі керування когнітивною картою

Основними двома класами таких задач є стабілізація когнітивної карти і утримання уже стабілізованої системи у стабільному стані нескінченно довго. У першому випадку необхідно синтезувати керування, яке приведе систему до стаціонарного стану - коли для деякого s $\Delta\boldsymbol{x}_{k+s}$ дорівнює $\mathbf{0}$ або знаходиться в околі цієї точки. У другому за мету ставиться утримання значення $\Delta\boldsymbol{x}$ в околі $\mathbf{0}$.

Структурна ідентичність формул (1.3) і (1.4) дозволяє застосовувати методи керування лінійними системами до цих задач без змін.

2.3 Класичний підхід до керування лінійними системами і когнітивними картами

Класичним підходом до керування лінійними системами і когнітивними картами є побудова дискретного контролера методами теорії керування. З цієї точки зору когнітивна карта розглядається як система із входом $\boldsymbol{u}\,(k)$ і

виходом $\boldsymbol{x}\,(k)$, що задається рівнянням

$$\left(I-(I+A)\,q^{-1}+Aq^{-2}\right)\boldsymbol{x}\,(k)=Bq^{-1}\boldsymbol{u}\,(k)\,, \tag{2.1}$$

де I - одинична матриця, q^{-1} - оператор зворотнього зсуву.

Лінійна система ж задається формулою

$$\left(I-Aq^{-1}\right)\boldsymbol{x}\,(k)=Bq^{-1}\boldsymbol{u}\,(k)\,. \tag{2.2}$$

Таким чином, система має матричну передаточну функцію

$$W(q^{-1})=\left(I-(I+A)\,q^{-1}+Aq^{-2}\right)^{-1}Bq^{-1} \qquad \text{або} \tag{2.3}$$

$$W(q^{-1})=\left(I-Aq^{-1}\right)^{-1}Bq^{-1} \tag{2.4}$$

для когнітивної карти і лінійної системи відповідно.

Це дозволяє застосовувати широкий спектр методів, розроблених на основі математичного апарату z-перетворення. За основну мету ставиться утворення лінійного рівняння, що дозволяє обчислити кожне наступне керування використовуючи попередні керування і стани системи (або їхні апроксимації) таким чином, щоб комбінація самої системи і такого контроллера була стабільною (у різних значеннях цього слова) і збігалася асимптотично до потрібного стану $\boldsymbol{x}^*$.

Недоліками такого підходу є те, що синтезований таким чином контролер наближатиме систему до бажаного стану асимптотично. Також проблематичним є те, що ці методи не дозволяють встановити обмеження на величини вектора керування $\boldsymbol{u}\,(k)$, а ці обмеження в реальних системах зазвичай присутні з об'єктивних причин.

2.4 Керування лінійними системами за допомогою прогнозуючої моделі

На відміну від підходу із застосуванням теорії керування, керування за допомогою прогнозуючої моделі полягає у знаходженні найкращої (згідно певного критерія) послідовності керувань серед можини усіх можливих керувань серед усіх можливих таких послідовностей, враховуючи обмеження (1.5).

Спільним у розглядаємих методах є те, що керування обчислюються одразу на s кроків уперед. Таким чином, рівняння (1.3) лінійної системи у детерміністичному випадку ($\boldsymbol{d}_k = 0$) перетворюється у

$$\Omega_s \boldsymbol{u}(k,s) = \boldsymbol{x}_{k+s} - A^s \boldsymbol{x}_k, \tag{2.5}$$

$$\boldsymbol{u}(k,s) = \begin{pmatrix} \boldsymbol{u}_k \\ \vdots \\ \boldsymbol{u}_{k+s-1} \end{pmatrix}, \tag{2.6}$$

$$\Omega_s = \begin{pmatrix} A^{s-1}B & A^{s-2}B & \dots & AB & B \end{pmatrix}. \tag{2.7}$$

У цьому варіанті системи нашою метою є синтез такої $\boldsymbol{u}(k,s)$, що $\boldsymbol{x}_{k+s} = \boldsymbol{x}^*$, тобто виконувалось рівняння

$$\Omega_s \boldsymbol{u}(k,s) = \boldsymbol{x}^* - A^s \boldsymbol{x}_k \tag{2.8}$$

при задоволенні обмежень (1.5).

У стохастичному ж випадку замість (2.5) ми отримуємо

$$\Omega_s \boldsymbol{u}(k,s) = \boldsymbol{x}_{k+s} - A^s \boldsymbol{x}_k - \sum_{i=k}^{k+s-1} A^{k+s-1-i} \boldsymbol{d}_i. \tag{2.9}$$

Є два основних підходи до керування системами за умови наявності випадкових збурень: робастний і ймовірністний. У наступних двох підрозділах буде показано з точки зору цих двох підходів, що у стохастичному випадку також слід використовувати керування, що задовольняє (2.8).

2.4.1 Робастне керування

Згідно (2.9), якщо на значення $\boldsymbol{d}$ немає обмежень, то система вважається некерованою, адже для будь якого наперед заданого керування $\boldsymbol{u}(k, s)$ множина можливих майбутніх станів у момент $k + s$ не обмежена. Якщо при при проектуванні системи керування відомі обмеження на збурення у вигляді

$$|d_{ki}| \leqslant d_{\max i}, \quad k \in \mathbb{Z}, \quad i \in \overline{1..n}, \tag{2.10}$$

де вектор $\boldsymbol{d}_{\max}$ складається із додатних чисел, то значення майбутнього стану $\boldsymbol{x}_{k+s}$ у момент $k + s$ обмежено многогранником із центром симетрії у точці $\Omega_s \boldsymbol{u}(k, s) + A^s \boldsymbol{x}_k$ згідно формули

$$\boldsymbol{x}_{k+s} = \Omega_s \boldsymbol{u}(k, s) + A^s \boldsymbol{x}_k + \sum_{i=k}^{k+s-1} A^{k+s-1-i} \boldsymbol{d}_i. \tag{2.11}$$

Як висновок, логічно синтезувати таке керування, щоби центр цього многогранника співпадав із цільовим станом $\boldsymbol{x}^*$. Керування з такою властивістю описується тим же рівнянням (2.8). У цій постановці задача належить до задач робастного (гарантованого) керування. "Гарантоване" воно у тому сенсі, що існує чітко окреслений окіл, у який стан системи гарантовано потрапить при застосуванні синтезованого керування і виконання обмежень на збурення (2.10).

2.4.2 Ймовірністне керування

Якщо про збурення нам відомо, що $\mathrm{E}[\boldsymbol{d}] = \mathbf{0}$ і що вони незалежні в сукупності, ми можемо виразити $\mathrm{E}\left[\boldsymbol{x}_{k+s}\right]$ як

$$\begin{aligned}\mathrm{E}[\boldsymbol{x}_{k+s}] &= \mathrm{E}\left[\Omega_s \boldsymbol{u}(k,s) + A^s \boldsymbol{x}_k + \sum_{i=k}^{k+s-1} A^{k+s-1-i}\boldsymbol{d}_i\right] = \\ &= \Omega_s \boldsymbol{u}(k,s) + A^s \boldsymbol{x}_k.\end{aligned} \tag{2.12}$$

Таким чином, ми отримуємо рівність

$$\Omega_s \boldsymbol{u}(k,s) = \mathrm{E}[\boldsymbol{x}_{k+s}] - A^s \boldsymbol{x}_k. \tag{2.13}$$

У такій постановці задачі природньо синтезувати таке керування, аби $\mathrm{E}\left[\boldsymbol{x}_{k+s}\right] = \boldsymbol{x}^*$. Це зводить задачу синтезу керування знову ж таки до розв'язання рівняння (2.8) при обмеженнях (1.5).

2.4.3 Оновлення керувань

Очевидно, що якщо ми можемо виміряти наступний стан системи після застосування частини наступних керувань, то природнім є перерахувати майбутні керування аби врахувати майбутні збурення. Це дозволяє на практиці підвищити точність керування в зашумленому середовищі, порівняно із застосуванням синтезованої послідовності керувань без змін.

2.4.4 Збіжність на дальній дистанції

Якщо для деякого конкретного горизонту s ціль $\boldsymbol{x}^*$ не є досяжною із поточного стану $\boldsymbol{x}_k$, хотілося б мати принаймні якесь правило, яке допомогло б нам обрати інший досяжний цільовий стан таким чином, щоб послідовність таких проміжних цілей або закінчувалась на стані, з якого початкова ціль досяжна, або хоча б аби ця послідовність збігалась до початкової цілі. Очевидно, що якщо ціль $\boldsymbol{x}^*$ досяжна за s_2 кроки, де $s_2 = ks, k \in \mathbb{N}, k \geqslant 2$, тоді така послідовність існує, але у загальному випадку не є очевидним, як її побудувати, окрім як просто збільшуючи s доки ми не знайдемо шукану послідовність керувань. Проте у випадку, коли $\boldsymbol{x}^* = \mathbf{0}$, ми можемо застосувати правило, що задовольняє умови теореми 1, 2, 3 або 4.

У наступних теоремах ми вважаємо, що $\|A\| \geqslant 1$, бо інакше стан системи буде сходитись до $\mathbf{0}$ навіть без застосування будь-яких керувань.

***Теорема* 1** ($1^{\text{ша}}$ умова збіжності)**.** *Нехай $\tilde{\boldsymbol{x}}_0 = \boldsymbol{x}_k$ і $\tilde{\boldsymbol{x}}_{i+1}$ - це наступна проміжна ціль, обрана, знаходячись у стані $\tilde{\boldsymbol{x}}_i$. Якщо $\|\tilde{\boldsymbol{x}}_{i+1}\| \leqslant \|\tilde{\boldsymbol{x}}_i\| - \delta, \delta > 0$, тоді стан системи досягне δ-околу $\mathbf{0}$ за скінченну кількість кроків. Якщо ця нерівність виконується доти, доки стан $\mathbf{0}$ не стає досяжним за s кроків, тоді система досягне такий стан за скінченну кількість кроків.*

Доведення.

$$\|\tilde{\boldsymbol{x}}_i\| \leqslant \|\tilde{\boldsymbol{x}}_0\| - i\delta \tag{2.14}$$

Як ми можемо бачити з (2.14), існує таке число i, що $\|\tilde{\boldsymbol{x}}_0\| - i\delta \leqslant \delta$.

□

***Теорема* 2** (Наслідок із теореми 1)**.** *Нехай $\tilde{\boldsymbol{x}}_0 = \boldsymbol{x}_k$ і $\tilde{\boldsymbol{x}}_{i+1}$ - це наступна проміжна ціль, обрана, знаходячись у стані $\tilde{\boldsymbol{x}}_i$. Якщо $\|A^s\tilde{\boldsymbol{x}}_i\| - \|\tilde{\boldsymbol{x}}_{i+1}\| \geqslant (\|A^s\| - 1)\|\tilde{\boldsymbol{x}}_i\| + \delta, \delta > 0$, тоді стан системи досягне δ-околу $\mathbf{0}$ за скінченну кількість кроків. Якщо ця нерівність виконується доти, доки стан $\mathbf{0}$ не стає досяжним за s кроків, тоді система досягне такий стан за скінченну кількість кроків.*

Доведення.

$$\|\tilde{\boldsymbol{x}}_{i+1}\| \leqslant \|A^s\tilde{\boldsymbol{x}}_i\| - (\|A^s\| - 1)\,\|\tilde{\boldsymbol{x}}_i\| - \delta \leqslant \|\tilde{\boldsymbol{x}}_i\| - \delta \tag{2.15}$$

Із (2.15) слідує, що вимоги теореми 1 задовольняються.

□

***Теорема* 3** (2[га] умова збіжності). *Нехай* $\tilde{\boldsymbol{x}}_0 = \boldsymbol{x}_k$ *і* $\tilde{\boldsymbol{x}}_{i+1}$ *- це наступна проміжна ціль, обрана, знаходячись у стані* $\tilde{\boldsymbol{x}}_i$*. Якщо* $\|A^s\|\|\tilde{\boldsymbol{x}}_i\| - \|\tilde{\boldsymbol{x}}_{i+1}\| \geqslant (\|A^s\| - 1)\,\|\tilde{\boldsymbol{x}}_0\| + \delta$, $\delta > 0$ *для кожного , для якого стан* **0** *не є досяжним із* $\tilde{\boldsymbol{x}}_i$ *за* s *кроків, тоді система досягне такий стан за скінченну кількість кроків.*

Доведення.

$$\begin{aligned}
\|\tilde{\boldsymbol{x}}_{i+1}\| \leqslant \|A^s\|\|\tilde{\boldsymbol{x}}_i\| - (\|A^s\| - 1)\,\|\tilde{\boldsymbol{x}}_0\| - \delta = \\
= \|A^s\|\,(\|\tilde{\boldsymbol{x}}_i\| - \|\tilde{\boldsymbol{x}}_0\|) + \|\tilde{\boldsymbol{x}}_0\| - \delta \leqslant \\
\leqslant \|A^s\|\,(\|A^s\|\|\tilde{\boldsymbol{x}}_{i-1}\| - \|A^s\|\|\tilde{\boldsymbol{x}}_0\| - \delta) + \|\tilde{\boldsymbol{x}}_0\| - \delta \leqslant \\
\leqslant \|A^s\|^2\,(\|\tilde{\boldsymbol{x}}_{i-1}\| - \|\tilde{\boldsymbol{x}}_0\|) + \|\tilde{\boldsymbol{x}}_0\| - (1 + \|A^s\|)\,\delta \leqslant \cdots \\
\cdots \leqslant \|A^s\|^{i+1}\,(\|\tilde{\boldsymbol{x}}_0\| - \|\tilde{\boldsymbol{x}}_0\|) + \|\tilde{\boldsymbol{x}}_0\| - \delta \sum_{k=0}^{i} \|A^s\|^k = \\
= \|\tilde{\boldsymbol{x}}_0\| - \delta \sum_{k=0}^{i} \|A^s\|^k
\end{aligned} \tag{2.16}$$

Як ми можемо бачити із (2.16), цей процес може тривати лише скінченну кількість кроків.

□

***Теорема* 4** (Наслідок з теореми 3). *Нехай* $\tilde{\boldsymbol{x}}_0 = \boldsymbol{x}_k$ *і* $\tilde{\boldsymbol{x}}_{i+1}$ *- це наступна проміжна ціль, обрана, знаходячись у стані* $\tilde{\boldsymbol{x}}_i$*. Якщо* $\|A^s\tilde{\boldsymbol{x}}_i\| - \|\tilde{\boldsymbol{x}}_{i+1}\| \geqslant (\|A^s\| - 1)\,\|\tilde{\boldsymbol{x}}_0\| + \delta$, $\delta > 0$ *для кожного , для якого стан* **0** *не є досяжним із* $\tilde{\boldsymbol{x}}_i$ *за* s *кроків, тоді система досягне такий стан за скінченну кількість кроків.*

Доведення.

$$\|A^s\tilde{\boldsymbol{x}}_i\| - \|\tilde{\boldsymbol{x}}_{i+1}\| \leqslant \|A^s\|\|\tilde{\boldsymbol{x}}_i\| - \|\tilde{\boldsymbol{x}}_{i+1}\| \tag{2.17}$$

Із (2.17) слідує, що вимоги теореми 3 задовольняються.

□

2.5 Висновки

У даному розділі було описано класи задач, розв'язання яких ставиться за мету у даній роботі. Також було описано основні принципи та підходи, що пропонуються, а саме: синтез керування на фіксованому горизонті з метою потрапляння стану системи у певну конкретну точку (або її окіл) у кінці цього горизонту.

Було доведено відповідні теореми, що гарантують збіжність стану лінійної системи до $\mathbf{0}$ (чи його околу) на дальній дистанції за скінченний час, якщо на кожному наступному горизонті довжиною s приводити систему в точку, норма якої менша за певне обмеження. Цей результат також корисний для стабілізації когнітивних карт, де до $\mathbf{0}$ треба привести величину $\Delta\boldsymbol{x}$.

3 МЕТОДИ РОЗВ'ЯЗАННЯ ЗАДАЧІ СПРЯМУВАННЯ У ЛІНІЙНІЙ СИСТЕМІ

3.1 Варіаційний метод

Основна ідея цього методу полягає в тому, щоби обчислити оптимальне керування не беручи до уваги будь-які обмеження, і лише потім вирішувати, що робити з отриманим результатом.

Керування обчислюється за наступною схемою: спочатку ми ділимо Ω_s по стовпцям на матриці $\Omega_s^{I_1}$ і $\Omega_s^{I_2}$, так що $\Omega_s^{I_1}$ - квадратна та оборотна, а $\Omega_s^{I_2}$ включала в себе усі інші стовпчики.

Ми можемо обчислити частини вектора $\boldsymbol{u}(k,s)$ ($\boldsymbol{u}_{I_1}(k,s)$ і $\boldsymbol{u}_{I_2}(k,s)$) як

$$\boldsymbol{u}_{I_1}(k,s) = \left[\Omega_s^{I_1} + \Omega_s^{I_2}\left(\Omega_s^{I_2}\right)^T\left(\Omega_s^{I_1}\right)^{-1}\right]^{-1}(\boldsymbol{x}^* - A^s\boldsymbol{x}_k) \qquad \text{і} \qquad (3.1)$$

$$\boldsymbol{u}_{I_2}(k,s) = \left(\Omega_s^{I_2}\right)^T\left(\Omega_s^{I_1}\right)^{-1}\boldsymbol{u}_{I_1}(k,s). \qquad (3.2)$$

Докладне доведення наведене у [2, розділ 3].

Урезультаті, ми отримуємо безпосередню формулу для обчислення послідовності керувань $\boldsymbol{u}(k,s)$ для кожного даного поточного стану $\boldsymbol{x}_k$ і цілі $\boldsymbol{x}^*$ у вигляді лінійного рівняння

$$\boldsymbol{u}(k,s) = K\left(\boldsymbol{x}^* - A^s\boldsymbol{x}_k\right), \qquad (3.3)$$

де K побудована згідно рівнянь (3.1) і (3.2).

Як уже було сказано, результат обчислений таким чином не обов'язково задовольняє обмеження (1.5). Але якщо $\boldsymbol{x}^* = \mathbf{0}$, тоді ми можемо застосувати певну стратегію збіжності, як це було описано у розділі 2.4.4. У ньому пропонується дві різні стратегії збіжності: лінійне наближення і проекційне наближення.

3.1.1 Стратегія лінійного наближення

Основна ідея лінійного наближення полягає у тому, щоби застосувати керування $\tilde{\boldsymbol{u}}(k,s)$, обчислене за формулою

$$\tilde{\boldsymbol{u}}(k,s) = \left(\min_{i \in \overline{1..sr}} \frac{u_{\max i}}{|u_i(k,s)|} \right) \cdot \boldsymbol{u}(k,s). \tag{3.4}$$

***Теорема* 5.** *Якщо* $\|\tilde{\boldsymbol{x}}_0\| < \frac{1}{\|A^s\|-1} \min_{j \in \overline{1..sr}} \frac{u_{\max j}}{\|K_j\|}$*, де* K_j *-це* j*-й рядок із* K*, тоді застосування керування* $\tilde{\boldsymbol{u}}(k,s)$*, обчисленого згідно* (3.4) *до детерміністичного варіанту системи дасть послідовність станів* $\tilde{\boldsymbol{x}}_i$*, які задовольняють теоркму 4.*

Доведення. Якщо $\boldsymbol{u}(k+is,s)$ задовольняє обмеження (1.5), тоді стан $\mathbf{0}$ може бути досягнутий із стану $\tilde{\boldsymbol{x}}_i$ за допомогою керування $\boldsymbol{u}(k+is,s)$. У іншому випадку, ми можемо вивести наступне.

Введемо позначення α_i як у виразі

$$\alpha_i = \min_{j \in \overline{1..sr}} \frac{u_{\max j}}{|u_j(k+is,s)|}. \tag{3.5}$$

Очевидно, що $0 < \alpha_i < 1$.

$$\Omega_s \boldsymbol{u}(k+is,s) = \mathbf{0} - A^s \tilde{\boldsymbol{x}}_i \tag{3.6}$$

$$\Omega_s \tilde{\boldsymbol{u}}(k+is,s) = \tilde{\boldsymbol{x}}_{i+1} - A^s \tilde{\boldsymbol{x}}_i \tag{3.7}$$

$$\begin{aligned} \tilde{\boldsymbol{x}}_{i+1} - A^s \tilde{\boldsymbol{x}}_i &= \Omega_s \left(\alpha_i \boldsymbol{u}(k+is,s) \right) = \\ &= \alpha_i \Omega_s \boldsymbol{u}(k+is,s) = \\ &= -\alpha_i A^s \tilde{\boldsymbol{x}}_i \end{aligned} \tag{3.8}$$

$$\tilde{\boldsymbol{x}}_{i+1} = (1-\alpha_i) A^s \tilde{\boldsymbol{x}}_i \tag{3.9}$$

Застосовуючи нерівність Коші-Буняковського-Шварца, ми можемо обмежити α_i як у формулі

$$\begin{aligned} \alpha_i = \min_{j \in \overline{1..sr}} \frac{u_{\max j}}{|K_j A^s \tilde{\boldsymbol{x}}_i|} \geqslant \\ \geqslant \min_{j \in \overline{1..sr}} \frac{u_{\max j}}{\|K_j\|} \cdot \frac{1}{\|A^s \tilde{\boldsymbol{x}}_i\|}. \end{aligned} \tag{3.10}$$

$$\begin{aligned} \|A^s \tilde{\boldsymbol{x}}_i\| - \|\tilde{\boldsymbol{x}}_{i+1}\| = \|A^s \tilde{\boldsymbol{x}}_i\| - (1-\alpha_i)\|A^s \tilde{\boldsymbol{x}}_i\| = \alpha_i \|A^s \tilde{\boldsymbol{x}}_i\| \geqslant \\ \geqslant \min_{j \in \overline{1..sr}} \frac{u_{\max j}}{\|K_j\|} \cdot \frac{1}{\|A^s \tilde{\boldsymbol{x}}_i\|} \|A^s \tilde{\boldsymbol{x}}_i\| = \\ = \min_{j \in \overline{1..sr}} \frac{u_{\max j}}{\|K_j\|} > (\|A^s\| - 1)\, \|\tilde{\boldsymbol{x}}_0\| \end{aligned} \tag{3.11}$$

Введемо позначення δ як у формулі

$$\delta = \min_{j \in \overline{1..sr}} \frac{u_{\max j}}{\|K_j\|} - (\|A^s\| - 1)\, \|\tilde{\boldsymbol{x}}_0\|. \tag{3.12}$$

Таким чином, ми можемо записати нерівність

$$\|A^s \tilde{\boldsymbol{x}}_i\| - \|\tilde{\boldsymbol{x}}_{i+1}\| \geqslant (\|A^s\| - 1)\, \|\tilde{\boldsymbol{x}}_0\| + \delta. \tag{3.13}$$

$\delta > 0$, тож умови теореми 4 задовольняються.

□

Як наслідок із варіаційного методу керування, ми отримуємо теорему 6.

***Теорема* 6.** *Для лінійної система в дискретному часі з багатьма входами та виходами, для кожного s існує ε-окіл стану* $\mathbf{0}$*, з якого цей стан є досяжним за s ітерацій за допомогою керувань, що задовольняють обмеження* (1.5)*. До того ж, $\varepsilon \geqslant \min_{i \in \overline{1..sr}} \frac{u_{\max i}}{\|K_i\|}$, де K_i -це i-й рядок матриці K.*

Доведення. Давайте спробуємо згенерувати s-крокову послідовність керувань $\boldsymbol{u}(k,s)$ із деякого стану $\boldsymbol{x}_k$ у стан $\mathbf{0}$ використовуючи варіаційний метод.

$$\boldsymbol{u}(k,s) = -KA^s\boldsymbol{x}_k \tag{3.14}$$

Використовуючи нерівність Коші-Буняковського-Шварца, ми отримуємо нерівність

$$|u_i(k,s)| \leqslant \|K_i\|\|\boldsymbol{x}_k\|, \quad i \in \overline{1..sr}. \tag{3.15}$$

Отож, виконання умови (3.16)

$$u_{\max i} \geqslant \|K_i\|\|\boldsymbol{x}_k\|, \quad i \in \overline{1..sr} \tag{3.16}$$

достатньо, аби керування (3.15) було допустимим.

Із (3.16) ми отримуємо

$$\|\boldsymbol{x}_k\| \leqslant \min_{i\in\overline{1..sr}} \frac{u_{\max i}}{\|K_i\|}. \tag{3.17}$$

□

У варіаційному методі задача синтезу керування розглядалась спочатку як задача без будь-яких обмежень на керування. Після отримання розв'язку в аналітичній формі множина досяжних станів штучно обмежувалась тими, які досяжні за допомогою керувань, синтезованих цим методом.

3.1.2 Стратегія проекційного наближення

Іншим підходом є вибір проекції стану **0** на множину станів, досяжних із $\tilde{\boldsymbol{x}}_i$ за s ітерацій за допомогою керувань, згенерованих варіаційним методом. Інакше кажучи, кожна проміжна ціль $\tilde{\boldsymbol{x}}_{i+1}$ є розв'язком задачі квадратичного програмування

$$\min_{\boldsymbol{x}^{\#}} \ \left\langle \boldsymbol{x}^{\#}, \boldsymbol{x}^{\#} \right\rangle \tag{3.18}$$

$$\text{за умови} \quad -u_{\max i} \leqslant K_i \left(\boldsymbol{x}^{\#} - A^s \tilde{\boldsymbol{x}}_i \right) \leqslant u_{\max i}, \quad i \in \overline{1..sr}. \tag{3.19}$$

Очевидно, що отриманий наступний проміжний стан $\tilde{\boldsymbol{x}}_{i+1}$ є таким, що його норма менше або дорівнює нормі проміжного стану, обчисленого з того самого поточного стану $\tilde{\boldsymbol{x}}_i$ за допомогою лінійного підходу. Таким чином, стратегія проекційного наближення також задовольняє умови теореми 4.

У цій роботі для розв'язку задачі (3.18), (3.19) використовується ітераційний розв'язувач `coneqp`. Його алгоритм описаний у [3]. Також, проміжний стан, обчислений за допомогою лінійного наближення використовується як відправна точка пошуку для розв'язувача.

3.2 Метод повного перебору

У випадках, коли апріорі відомо, що цільовий стан $\boldsymbol{x}^*$ належить області досяжності, заданої початковим станом $\boldsymbol{x}k$, задача спрямування має бути розв'язана як оптимізаційна задача з урахуванням обмежень на керування. Вона може бути записана, наприклад, як задача квадратичного програмування з обмеженнями у вигляді рівностей (2.8) і нерівностей (1.5). Необхідні і достатні умови оптимальності слідують безпосередньо з теореми Куна-Такера [1]. На основі цих умов записується система рівнянь, згідно яких пропонує-

ться ітеративна схема знаходження розв'язку.

На відміну від варіаційного методу, тут ми не будемо звужувати множину досяжності для $\boldsymbol{x}^*$ заради зручності. Натомість, ми виконаємо пошук оптимального керування (у розумінні, заданому формулою (3.21)) повним перебором по множині підзадач, отриманих із основної задачі шляхом додавання деяких штучних обмежень.

3.2.1 Означення множини допустимих керувань

Перш за все, нам необхідне формальне означення множини керувань $\boldsymbol{u}(k,s)$, які задовольняють обмеження і приводять систему із початкового стану $\boldsymbol{x}_k$ у стан $\boldsymbol{x}^*$ за s кроків. Очевидно, це означення - це об'єднання (2.8) і (1.5). Воно може бути записане як

$$U_{\text{per}}(\boldsymbol{x}^*, \boldsymbol{u}_{\max}) = \{\boldsymbol{u}(k,s) : \quad \begin{cases} \Omega_s \cdot \boldsymbol{u}(k,s) = \boldsymbol{x}^* - A^s \boldsymbol{x}_k \\ u_j(k,s) \leqslant u_{\max j}, \quad j \in \overline{1..sr} \\ -u_j(k,s) \leqslant u_{\max j}, \quad j \in \overline{1..sr} \end{cases} \}, \tag{3.20}$$

де $\boldsymbol{u}_{\max} = \left(\boldsymbol{u}_{\max}^T, \boldsymbol{u}_{\max}^T, \ldots, \boldsymbol{u}_{\max}^T\right)^T$.

3.2.2 Пошук оптимального керування

Нехай множина $U_{\text{per}}(\boldsymbol{x}^*, \boldsymbol{u}_{\max})$ не порожня. У цьому випадку нам треба обрати оптимальне керування із цієї множини. Ми називатимемо "оптимальним" керування, задане як

$$\boldsymbol{u}^*(k,s) \stackrel{\text{def}}{=} \underset{\boldsymbol{u}(k,s) \in U_{\text{per}}(\boldsymbol{x}^*, \boldsymbol{u}_{\max})}{\arg\min} \frac{1}{2} \sum_{j=1}^{sr} u_j^2(k,s). \tag{3.21}$$

Таким чином мим маємо задачу квадратичного програмування з обмеженнями у вигляді лінійних рівнянь і нестрогих нерівностей. Функція Лагранжа для цієї задачі приймає вигляд

$$L(\boldsymbol{u}(k,s),\boldsymbol{\lambda},\boldsymbol{v}^{(-)},\boldsymbol{v}^{(+)}) = \frac{1}{2}\sum_{j=1}^{sr} u_j^2(k,s) + \\ + \left\langle \boldsymbol{\lambda}, \Omega_s^j \cdot \boldsymbol{u}(k,s) - (\boldsymbol{x}^* - A^s\boldsymbol{x}_k)\right\rangle + \\ + \left\langle \boldsymbol{v}^{(-)}, -\boldsymbol{u}(k,s) - \boldsymbol{u}_{\max}\right\rangle + \\ + \left\langle \boldsymbol{v}^{(+)}, \boldsymbol{u}(k,s) - \boldsymbol{u}_{\max}\right\rangle, \quad (3.22)$$

де $\boldsymbol{\lambda}$, $\boldsymbol{v}^{(-)}$ і $\boldsymbol{v}^{(+)}$ - множники Лагранжа, а Ω_s^j - j-й стовпець матриці Ω_s.

Згідно теореми Куна-Такера, розв'язок $\boldsymbol{u}^*(k,s)$ задачі (3.20), (3.21) та відповідні множники Лагранжа $\boldsymbol{\lambda}^*$, $\boldsymbol{v}^{(-)*}$ і $\boldsymbol{v}^{(+)*}$ задовольняють системі

$$\begin{cases} \dfrac{\partial L}{\partial \boldsymbol{u}(k,s)}(\boldsymbol{u}^*(k,s),\boldsymbol{\lambda}^*,\boldsymbol{v}^{(-)*},\boldsymbol{v}^{(+)*}) = 0 \\ \Omega_s \cdot \boldsymbol{u}^*(k,s) = \boldsymbol{x}^* - A^s\boldsymbol{x}_k \\ v_j^{(-)*}(-u_j^*(k,s) - u_{\max j}) = 0, & j \in \overline{1..sr} \\ v_j^{(+)*}(u_j^*(k,s) - u_{\max j}) = 0, & j \in \overline{1..sr} \\ v_j^{(-)*} \geqslant 0, & j \in \overline{1..sr} \\ v_j^{(+)*} \geqslant 0, & j \in \overline{1..sr} \end{cases}. \quad (3.23)$$

Перший вираз із (3.23) може бути записаний також як

$$u_j^*(k,s) + \left\langle \boldsymbol{\lambda}^*, \Omega_s^j \right\rangle = v_j^{(-)*} - v_j^{(+)*}, \quad j \in \overline{1..sr} \qquad \text{або} \qquad (3.24)$$

$$\boldsymbol{u}^*(k,s) + \Omega_s^T\boldsymbol{\lambda}^* = \boldsymbol{v}^{(-)*} - \boldsymbol{v}^{(+)*}. \qquad (3.25)$$

3.2.3 Підмножини індексів вектора керування

Задамо підмножини

$$I_{\min} = \{j \in \overline{1..sr} : \ v_j^{(-)*} > 0\}, \tag{3.26}$$
$$I_{\max} = \{j \in \overline{1..sr} : \ v_j^{(+)*} > 0\} \quad \text{і} \tag{3.27}$$
$$I_0 \quad = \{j \in \overline{1..sr} : \ v_j^{(-)*} = 0, \ v_j^{(+)*} = 0\} \tag{3.28}$$

монжини індексів $\overline{1..sr}$.

Із системи (3.23) ми можемо отримати вирази

$$\forall j \in I_{\min} \qquad u_j^*(k,s) = -u_{\max j}, \qquad v_j^{(+)*} = 0 \quad \text{і} \tag{3.29}$$
$$\forall j \in I_{\max} \qquad u_j^*(k,s) = \quad u_{\max j}, \qquad v_j^{(-)*} = 0. \tag{3.30}$$

Варто зазначити, що у нас інформації щодо того, чи $u_j^*(k,s)$ знаходиться усередині відрізку $[-u_{\max j}, \ u_{\max j}]$ для $j \in I_0$.

З цього витікає, що $I_{\min} \cap I_{\max} = \varnothing$, $I_0 \cap I_{\min} = \varnothing$, $I_0 \cap I_{\max} = \varnothing$, а також $I_0 \cup I_{\min} \cup I_{\max} = \overline{1..sr}$.

Таким чином, ми можемо роздіити (3.25) по рядкам використовуючи множини I_0, $I_{\min}$ та $I_{\max}$. Але аби описати таке розбиття лаконічно, нам потрібно ввести специфічну нотацію.

Аби описати матрицю або вектор, що складається з рядків або елементів, відповідних індексам у певній множині, ми записуватимему цю множину у нижньому індексі вихідної матриці чи вектора.

Наприклад, $\boldsymbol{u}_{I_0}^*(k,s)$ - це вектор, що складається з елементів із $\boldsymbol{u}^*(k,s)$, що відповідають індексам у I_0, а $\left(\Omega_s^T\right)_{I_0}$ - це матриця, яка складається з рядків матриці Ω_s^T, що відповідають тим самим індексам.

Також, аби описати матрицю, що складається зі стовпців відповідних до індексів у заданій множині, ми вказуватимемо цю множину у верхньому

індексі. Наприклад, $\Omega_s^{I_0}$ - це матриця, що складається зі стовпців, узятих із Ω_s за індексами з множини I_0.

До речі, таким чином очевидним є вираз $\left(\Omega_s^T\right)_{I_0} = \left(\Omega_s^{I_0}\right)^T$.

І нарешті, розбиваючи систему рівнянь (3.25) ми отримуємо

$$\begin{cases} \boldsymbol{u}^*_{I_0}(k,s) + \left(\Omega_s^T\right)_{I_0} \boldsymbol{\lambda}^* = \mathbf{0} \\ \boldsymbol{v}^{(+)*}_{I_{\min}} = \mathbf{0}; \quad \boldsymbol{u}^*_{I_{\min}}(k,s) + \left(\Omega_s^T\right)_{I_{\min}} \boldsymbol{\lambda}^* = \quad \boldsymbol{v}^{(-)*} > \mathbf{0} \\ \boldsymbol{v}^{(+)*}_{I_{\max}} = \mathbf{0}; \quad \boldsymbol{u}^*_{I_{\max}}(k,s) + \left(\Omega_s^T\right)_{I_{\max}} \boldsymbol{\lambda}^* = -\boldsymbol{v}^{(+)*} < \mathbf{0} \end{cases} . \tag{3.31}$$

3.2.4 Застосування штучних обмежень

Аби продовжити розв'язувати цю задачу, нам необхідно зробити припущення щодо вмісну I_0, $I_{\min}$ та $I_{\max}$. Усього є 3^{sr} варіанти їх вмісту, і щоби знайти розв'язок, нам необхідно перевіряти їх послідовно, доки не знайдемо правильну комбінацію.

Усього є три основних групи варіантів із фундаментально різними способами знаходження розв'язку:

а) ті, у яких кількість індексів у I_0 рівна розмірності $\boldsymbol{\lambda}^*$, тобто $\operatorname{card} I_0 = \dim \boldsymbol{\lambda}^* = n$, що означає, що перший вираз у (3.31) - детермінована система лінійних рівнянь.

б) ті, у яких кількість індексів у I_0 строго більша, ніж розмірність $\boldsymbol{\lambda}^*$, тобто $\operatorname{card} I_0 > \dim \boldsymbol{\lambda}^* = n$, що означає, що перший вираз у (3.31) - перевизначена система лінійних рівнянь.

в) ті, у яких кількість індексів у I_0 строго меньша, ніж розмірність $\boldsymbol{\lambda}^*$, тобто $\operatorname{card} I_0 < \dim \boldsymbol{\lambda}^* = n$, що означає, що перший вираз у (3.31) - недовизначена система лінійних рівнянь.

Щоб знайти оптимальне керування, нам необхідно виконати пошук серед усіх можливих варіантів вмісту $I_{\min}$, $I_{\max}$ та I_0 і спробувати розв'язати ці проблеми для кожного випадку окремо. Це вимагає зного об'єму обчислень,

але з сучасними обчислювальними технологіями це все ще можливо.

3.2.5 Випадок детермінованої системи рівнянь

У випадку детермінованої системи рівнянь ми можемо записати рівність

$$\boldsymbol{\lambda}^* = -\left(\left(\Omega_s^T\right)_{I_0}\right)^{-1} \boldsymbol{u}^*_{I_0}(k,s). \tag{3.32}$$

для $\boldsymbol{\lambda}^*$ із (3.31), якщо матриця $\left(\Omega_s^T\right)_{I_0}$ - оборотна. Інакше, ми можемо зробити висновок, що цей конкретний варіант I_0 не дає розв'язку задачі.

Застосовуючи (3.32) до нерівностей у (3.31), ми отримуємо

$$\boldsymbol{u}^*_{I_{\min}}(k,s) - \left(\Omega_s^T\right)_{I_{\min}} \left(\left(\Omega_s^T\right)_{I_0}\right)^{-1} \boldsymbol{u}^*_{I_0}(k,s) > 0 \quad \text{і} \tag{3.33}$$

$$\boldsymbol{u}^*_{I_{\max}}(k,s) - \left(\Omega_s^T\right)_{I_{\max}} \left(\left(\Omega_s^T\right)_{I_0}\right)^{-1} \boldsymbol{u}^*_{I_0}(k,s) < 0. \tag{3.34}$$

Як ми уже знаємо, $\boldsymbol{u}^{I_{\min}}_{=}(k,s) - \boldsymbol{u}_{\max I_{\min}}$ і $\boldsymbol{u}^{I_{\max}}_{=}(k,s)\boldsymbol{u}_{\max I_{\max}}$. Тому, (3.33) і (3.34) для зручності можуть бути перетворені на

$$\left(\Omega_s^T\right)_{I_{\min}} \left(\left(\Omega_s^T\right)_{I_0}\right)^{-1} \boldsymbol{u}^*_{I_0}(k,s) < -\boldsymbol{u}_{\max I_{\min}} \quad \text{і} \tag{3.35}$$

$$\left(\Omega_s^T\right)_{I_{\max}} \left(\left(\Omega_s^T\right)_{I_0}\right)^{-1} \boldsymbol{u}^*_{I_0}(k,s) > \quad \boldsymbol{u}_{\max I_{\max}}. \tag{3.36}$$

Якщо ми розіб'ємо другу рівність із (3.23) використовуючи $I_{\min}$, $I_{\max}$ та I_0, ми отримаємо

$$\Omega_s^{I_0} \cdot \boldsymbol{u}^*_{I_0}(k,s) = (\boldsymbol{x}^* - A^s \boldsymbol{x}_k) - \Omega_s^{I_{\min}} \boldsymbol{u}^*_{I_{\min}}(k,s) - \Omega_s^{I_{\max}} \boldsymbol{u}^*_{I_{\max}}(k,s). \tag{3.37}$$

Таким чином, ми можемо безпосередньо обчислити $\boldsymbol{u}^*_{I_0}(k,s)$ за формулою

$$\boldsymbol{u}^*_{I_0}(k,s) = \left(\Omega_s^{I_0}\right)^{-1}\left((\boldsymbol{x}^* - A^s\boldsymbol{x}_k) + \Omega_s^{I_{\min}}\boldsymbol{u}_{\max I_{\min}} - \Omega_s^{I_{\max}}\boldsymbol{u}_{\max I_{\max}}\right). \qquad (3.38)$$

Якщо цей результат задовольняє (3.35) і (3.36), тоді це є оптимальне керування. Інакше, нам треба спробувати інший варіант вмісту $I_{\min}$, $I_{\max}$ та I_0.

3.2.6 Випадок перевизначеної системи рівнянь

У цьому випадку нам необхідно розділити I_0 на підмножини $I_{0,1}$ та $I_{0,2}$ так, щоб $\operatorname{card} I_{0,1} = \dim \boldsymbol{\lambda}^*$, $I_{0,1} \cup I_{0,2} = I_0$, $I_{0,1} \cap I_{0,2} = \varnothing$ і $\Omega_s^{I_{0,1}}$ була оборотною. Також, бажаним було б обрати такий варіант $I_{0,1}$, який дає матрицю $\Omega_s^{I_{0,1}}$ з найменшим числом визначеності, аби обчислення були більш точними.

Шляхом додаткового розділення, ми отримуємо

$$\boldsymbol{u}^*_{I_{0,1}}(k,s) + \left(\Omega_s^T\right)_{I_{0,1}}\boldsymbol{\lambda}^* = \boldsymbol{0} \quad \text{і} \qquad (3.39)$$

$$\boldsymbol{u}^*_{I_{0,2}}(k,s) + \left(\Omega_s^T\right)_{I_{0,2}}\boldsymbol{\lambda}^* = \boldsymbol{0} \qquad (3.40)$$

із першого виразу системи (3.31).

Таким чином, подібно до попереднього випадку, ми отримуємо рівняння

$$\boldsymbol{\lambda}^* = -\left(\left(\Omega_s^T\right)_{I_{0,1}}\right)^{-1}\boldsymbol{u}^*_{I_{0,1}}(k,s). \qquad (3.41)$$

Застосувавши (3.41) до (3.40), ми отримуємо вираз

$$\boldsymbol{u}^*_{I_{0,2}}(k,s) = \left(\Omega_s^T\right)_{I_{0,2}}\left(\left(\Omega_s^T\right)_{I_{0,1}}\right)^{-1}\boldsymbol{u}^*_{I_{0,1}}(k,s). \qquad (3.42)$$

І як і в попередньому випадку, ми отримуємо нерівності

$$\boldsymbol{u}^*_{I_{\min}}(k,s) - \left(\Omega_s^T\right)_{I_{\min}} \left(\left(\Omega_s^T\right)_{I_{0,1}}\right)^{-1} \boldsymbol{u}^*_{I_{0,1}}(k,s) > \mathbf{0} \quad \text{і} \tag{3.43}$$

$$\boldsymbol{u}^*_{I_{\max}}(k,s) - \left(\Omega_s^T\right)_{I_{\max}} \left(\left(\Omega_s^T\right)_{I_{0,1}}\right)^{-1} \boldsymbol{u}^*_{I_{0,1}}(k,s) < \mathbf{0}, \tag{3.44}$$

які також можуть бути записані як

$$\left(\Omega_s^T\right)_{I_{\min}} \left(\left(\Omega_s^T\right)_{I_{0,1}}\right)^{-1} \boldsymbol{u}^*_{I_{0,1}}(k,s) < -\boldsymbol{u}_{\max I_{\min}} \quad \text{і} \tag{3.45}$$

$$\left(\Omega_s^T\right)_{I_{\max}} \left(\left(\Omega_s^T\right)_{I_{0,1}}\right)^{-1} \boldsymbol{u}^*_{I_{0,1}}(k,s) > \quad \boldsymbol{u}_{\max I_{\max}}, \tag{3.46}$$

а також рівність

$$\Omega_s^{I_{0,1}} \cdot \boldsymbol{u}^*_{I_{0,1}}(k,s) + \Omega_s^{I_{0,2}} \cdot \boldsymbol{u}^*_{I_{0,2}}(k,s) = (\boldsymbol{x}^* - A^s \boldsymbol{x}_k) - \Omega_s^{I_{\min}} \boldsymbol{u}^*_{I_{\min}}(k,s) - \\ - \Omega_s^{I_{\max}} \boldsymbol{u}^*_{I_{\max}}(k,s). \tag{3.47}$$

Шляхом заміни $\boldsymbol{u}^*_{I_{0,2}}(k,s)$ у (3.47) за допомогою виразу (3.42), ми отримуємо вираз

$$\Omega_s^{I_{0,1}} \cdot \boldsymbol{u}^*_{I_{0,1}}(k,s) + \Omega_s^{I_{0,2}} \cdot \left(\Omega_s^T\right)_{I_{0,2}} \left(\left(\Omega_s^T\right)_{I_{0,1}}\right)^{-1} \cdot \boldsymbol{u}^*_{I_{0,1}}(k,s) = \\ = (\boldsymbol{x}^* - A^s \boldsymbol{x}_k) - \Omega_s^{I_{\min}} \boldsymbol{u}^*_{I_{\min}}(k,s) - \Omega_s^{I_{\max}} \boldsymbol{u}^*_{I_{\max}}(k,s), \tag{3.48}$$

який також можна записати як

$$\left(\Omega_s^{I_{0,1}} + \Omega_s^{I_{0,2}} \cdot \left(\Omega_s^T\right)_{I_{0,2}} \left(\left(\Omega_s^T\right)_{I_{0,1}}\right)^{-1}\right) \cdot \boldsymbol{u}^*_{I_{0,1}}(k,s) = \\ = (\boldsymbol{x}^* - A^s \boldsymbol{x}_k) - \Omega_s^{I_{\min}} \boldsymbol{u}^*_{I_{\min}}(k,s) - \Omega_s^{I_{\max}} \boldsymbol{u}^*_{I_{\max}}(k,s). \tag{3.49}$$

Таким чином, ми можемо записати безпосередню формулу для обчислення $\boldsymbol{u}^*_{I_{0,1}}(k,s)$ у вигляді

$$\boldsymbol{u}^*_{I_{0,1}}(k,s) = \left(\Omega_s^{I_{0,1}} + \Omega_s^{I_{0,2}} \cdot \left(\Omega_s^T\right)_{I_{0,2}} \left(\left(\Omega_s^T\right)_{I_{0,1}}\right)^{-1}\right)^{-1} \cdot \\ \cdot \left((\boldsymbol{x}^* - A^s \boldsymbol{x}_k) - \Omega_s^{I_{\min}} \boldsymbol{u}^*_{I_{\min}}(k,s) - \Omega_s^{I_{\max}} \boldsymbol{u}^*_{I_{\max}}(k,s)\right). \quad (3.50)$$

Якщо цей результат задовольняє (3.45) і (3.46), а також (1.5), то він є оптимальним керуванням. Інакше, нам треба спробувати інший варіант вмісту $I_{\min}$, $I_{\max}$ та I_0.

3.2.7 Випадок недовизначеної системи рівнянь

Якщо перша частина рівняння (3.31) недовизначена, то ми не можемо виразити $\boldsymbol{\lambda}^*$ використовуюи $\boldsymbol{u}^*_{I_0}(k,s)$, як це було зроблено у (3.32) і (3.41). Таким чином, найкраще, що ми можем зробити з цим рівнянням, це

$$\boldsymbol{u}^*_{I_0}(k,s) = -\left(\Omega_s^T\right)_{I_0} \boldsymbol{\lambda}^*. \quad (3.51)$$

Отож, аби знайти $\boldsymbol{u}^*_{I_0}(k,s)$, нам спочатку необхідно знайти $\boldsymbol{\lambda}^*$. Як і у попередніх двох випадках, ми можемо розділити друге рівняння у системі (3.23) використовуючи $I_{\min}$, $I_{\max}$ та I_0 і отримати таким чином рівняння (3.37). Застосувавши до нього (3.51), ми отримаємо вираз

$$-\Omega_s^{I_0}\left(\Omega_s^T\right)_{I_0} \boldsymbol{\lambda}^* = (\boldsymbol{x}^* - A^s \boldsymbol{x}_k) - \Omega_s^{I_{\min}} \boldsymbol{u}^*_{I_{\min}}(k,s) - \Omega_s^{I_{\max}} \boldsymbol{u}^*_{I_{\max}}(k,s), \quad (3.52)$$

який такод можна записати як

$$-\Omega_s^{I_0}\left(\Omega_s^T\right)_{I_0}\boldsymbol{\lambda}^* = (\boldsymbol{x}^* - A^s\boldsymbol{x}_k) + \Omega_s^{I_{\min}}\boldsymbol{u}_{\max I_{\min}} - \Omega_s^{I_{\max}}\boldsymbol{u}_{\max I_{\max}}. \tag{3.53}$$

Було би прекрасно, якби матриця $\Omega_s^{I_0}\left(\Omega_s^T\right)_{I_0}$ була оборотною, але її ранг не повний. Таким чином, у загальному випадку, система рівнянь (3.53) не має розв'язку.

3.3 Метод найкращої цілі

Метод найкращої цілі був запропонований у [2, розділ 5]. У цьому методі ми розв'язуємо задачу

$$\begin{aligned}\min J(\boldsymbol{u}(k,s)) = \frac{1}{2}\big(&\langle\Omega_s\boldsymbol{u}(k,s) + A^s\boldsymbol{x}_k - \boldsymbol{x}^*, \Omega_s\boldsymbol{u}(k,s) + A^s\boldsymbol{x}_k - \boldsymbol{x}^*\rangle + \\ &+ \alpha\,\langle\boldsymbol{u}(k,s), \boldsymbol{u}(k,s)\rangle\big)\end{aligned} \tag{3.54}$$

з обмеженями (1.5) багато разів, щоразу зменшуючи значення додатнього коефіцієнта α.

Ми можемо записати множину досяжних X_{reach} станів як

$$\begin{aligned}X_{\text{reach}} = \big\{\boldsymbol{x}\ : &\boldsymbol{x} = \Omega_s\boldsymbol{u} + A^s\boldsymbol{x}_k, \\ &-u_{\max i} \leqslant u_i \leqslant u_{\max i},\ i \in \overline{1..sr}\big\}\end{aligned} \tag{3.55}$$

$$\boldsymbol{x}^{\#} = \arg\min_{\boldsymbol{x}\in X_{\text{reach}}} \|\boldsymbol{x} - \boldsymbol{x}^*\|. \tag{3.56}$$

Теоретично, можна сказати що твердження

$$\Omega_s \boldsymbol{u}(k,s) + A^s \boldsymbol{x}_k \xrightarrow[\alpha \to 0]{} \boldsymbol{x}^{\#} \qquad \text{і} \qquad (3.57)$$

$$\boldsymbol{u}(k,s) \xrightarrow[\alpha \to 0]{} \arg \min_{\Omega_s \boldsymbol{u} + A^s \boldsymbol{x}_k = \boldsymbol{x}^{\#}} \langle \boldsymbol{u}, \boldsymbol{u} \rangle \qquad (3.58)$$

справедливі, тому ми можемо таким чином з певною точністю знайти керування, що в момент часу $k+s$ приведе детерміністичну систему так близько до точки $\boldsymbol{x}^*$, наскільки це можливо. У загальному випадку не відомо, чи цей процес дасть послідовність проміжних цілей, яка зійдеться до початкової цільової точки $\boldsymbol{x}^*$.

Нащастя, у разі якщо $\boldsymbol{x}^* = \mathbf{0}$, тоді таким чином ми знайдемо таке керування $\boldsymbol{u}(k,s)$, що приведе стан системи навіть ближче до $\mathbf{0}$ (або принаймні не далі), аніж лінійний підхід варіаційного методу, у випадку коли ми ще не можемо досягнути $\mathbf{0}$ за s ітерацій. Таким чином ми можемо гарантувати, що у цьому разі послідовність проміжних цілей зійдеться до $\mathbf{0}$.

Хотілося б підкреслити, що X_{reach} - це множина станів, досяжних із поточного стану $\boldsymbol{x}_k$, загалом, без урахування обмежень будь-яких конкретних алгоритмів синтезу керування. Таким чином, вона є надмножиною множини станів, досяжних за допомогою керувань, згенерованих варіаційним методом. Як наслідок, цей метод також дає нам проміжні цілі, які лежать не далі, аніж згенеровані проекційним підходом варіаційного методу. Таким чином, використання $\boldsymbol{x}^{\#}$ обчисленого таким чином як наступного проміжного стану також задовольняє теорему (4).

Але на практиці ми не можемо обчислити розв'язок задачі (3.54), (1.5) для будь-яких малих α через обмеження на точність обчислень із плаваючою комою. Отож, найкраще, що ми можемо зробити, це припинити процес ітераційного зменшення α, як тільки наступне отримане керування $\boldsymbol{u}(k,s)$ приводить систему далі від $\mathbf{0}$, аніж попередній результат. Таким чином ми синтезуємо нову послідовність керувань із достатньою точністю.

У якості компромісу між точністю та швидкістю обчислень, ми також можемо заздалегідь обрати достатньо мале значення α і розв'язувати задачу (3.54), (1.5) тільки з ним.

3.3.1 Метод розв'язку шляхом пошуку підмножин індексів

У [2, розділ 5] було запропоновано підхід до пошуку розбиття індексів для задачі (3.58).

Основна ідея цього підходу полягає в наступному. Спочатку ми намагаємось просто вгадати правильне розбиття множини індексів $\overline{1..sr}$ на три підмножини I_0, $I_{\min}$ і $I_{\max}$. Потім ми намагаємось обчислити керування $\boldsymbol{u}(k,s)$ згідно формулам

$$u_i(k,s) = -u_{\max i}, \quad i \in I_{\min}, \tag{3.59}$$

$$u_i(k,s) = \quad u_{\max i}, \quad i \in I_{\max}, \tag{3.60}$$

$$\begin{aligned} \boldsymbol{w}_{I_{\min}\cup I_{\max}} = &\left(\left(\Omega^{-1}\right)_{I_{\min}\cup I_{\max}}^{I_{\min}\cup I_{\max}} \right)^{-1} \cdot \\ &\cdot \left(\boldsymbol{u}_{I_{\min}\cup I_{\max}}(k,s) - \left(\Omega^{-1}\right)_{I_{\min}\cup I_{\max}} \cdot \boldsymbol{x}^{**} \right), \end{aligned} \tag{3.61}$$

$$\boldsymbol{u}_{I_0}(k,s) = \left(\Omega^{-1}\right)_{I_0} \cdot \boldsymbol{x}^{**} + \left(\Omega^{-1}\right)_{I_0}^{I_{\min}\cup I_{\max}} \boldsymbol{w}_{I_{\min}\cup I_{\max}}, \tag{3.62}$$

де $\Omega = \Omega_s^T \Omega_s + \alpha E$ і $\boldsymbol{x}^{**} = \Omega_s^T \left(\boldsymbol{x}^* - A^s \boldsymbol{x}_k\right)$.

Якщо отримане керування задовольняє обмеження (1.5) і $\boldsymbol{w}_{I_{\min}} > 0$, $\boldsymbol{w}_{I_{\max}} < 0$ (те, що нерівності строгі, суттєво), тоді ми знайшли розв'язок задачі (3.54) з обмеженнями (1.5). Інакше нам потрібно спробувати інше розбиття множини індексів.

Було б прекрасно мати якесь правило для вибору наступного розбиття множини індексів, яке б базувалося на результатах попередніх спроб і яке б привело нас до правильного розбиття швидше, аніж перебір усіх варіантів послідовно або у випадковому порядку.

3.3.2 Правило послідовного обходу

У [2, розділ 6] було запропоноване таке правило, але воно має недолік: для деяких таких задач воно заведе пошук у цикл. Отож, якщо ми хочемо застосовувати його на практиці, нам необхідно буде реалізувати перевірку на такі цикли. Якщо було виявлено цикл у пошуку, нам потрібно обрати наступне розбиття множини індексів якимось іншим способом - наприклад, випадковим чином.

Нам також потрібно обрати розбиття, яке перевірятимемо першим. Пропонується як таке узяти розбиття $I_0 = \overline{1..sr}$, $I_{\min} = \varnothing$, $I_{\max} = \varnothing$, бо якщо $A^s \boldsymbol{x}_k$ знаходиться у певному околі $\boldsymbol{x}^*$, тоді розв'язок буде відповідати цьому розбиттю.

Ідея правила послідовного обходу полягає в тому, щоби обрати наступне розбиття використовуючи обчислення, виконані для попереднього. Якщо точніше, наступне розбиття генерується таким чином:

- переносимо із I_0 у $I_{\min}$ такі $i \in I_0$, для яких $u_i(k,s) < -u_{\max i}$ (бо таким чином для них порушуються обмеження (1.5));
- переносимо із I_0 у $I_{\max}$ такі $i \in I_0$, для яких $u_i(k,s) > u_{\max i}$ (бо таким чином для них також порушуються обмеження (1.5));
- переносимо із $I_{\min}$ у I_0 такі $i \in I_{\min}$, для яких $w_i \leqslant 0$;
- переносимо із $I_{\max}$ у I_0 такі $i \in I_{\max}$, для яких $w_i \geqslant 0$.

Цей алгоритм має одну важливу властивість: якщо ми спробуємо за допомогою нього згенерувати наступне розбиття із того розбиття, яке відповідає розв'язку, тоді ми отримаємо те саме розбиття.

Як уже було сказано раніше, цей алгоритм може приводити до циклів пошуку. Таким чином, якщо ми побудуємо направлений граф обходу розбиттів, він буде складатися з однієї чи більше компонент зв'язності. Одна з нах, очевидно, буде включати розв'язок, і ми досягнемо цього розв'язку, якщо почнемо пошук із будь-якої з вершин із цієї компоненти. Пошук, що почнеться з вершини із будь-якої іншої компоненти, потрапить у цикл.

Згідно з результатами тестових запусків із випадково згенерованими за-

дачами, ситуація, коли процес пошуку потрапляє у цикл, зустрічається відносно не так часто. Але такі ситуації не настільки рідкісні, щоб можна було забути про їх існування. Результати тестових запусків наведено у табл. 3.1.

Таблиця 3.1 — Розподіл пошукових ситуацій для випадково згенерованих задач при застосуванні алгоритму послідовного обходу.

n	r	s	Тривіальні	Нетривіальні	З першої спроби	Цикли	Усього
3	3	1	2117	7847	9964	36	10000
4	3	2	6519	3059	9578	422	10000
5	3	2	4646	4989	9635	365	10000
6	3	2	2676	7243	9919	81	10000
7	3	3	6125	3411	9536	464	10000

У цій таблиці стовпчики містять наступне:

- n - розмірність стану ;
- r - розмірність керуючого вектора ;
- s - кількість згенерованих ітерацій керування ;
- Тривіальні - кількість зразків з розв'язками, що відповідають розбиттю $I_0 = \overline{1..sr}$, $I_{\min} = \varnothing$, $I_{\max} = \varnothing$;
- Нетривіальні - кількість зразків, для яких розв'язок було знайдено без потрапляння в цикл, за виключенням "Тривіальних" випадків ;
- З першої спроби - загальна кількість зразків, для яких розв'язок було знайдено без потрапляння у цикл ("Тривіальні" і "Нетривіальні" випадки) ;
- Цикли - кількість зразків, для яких процес пошуку потрапив у цикл ;
- Усього - загальна кількість розглянутих задач із такими значеннями коефіцієнтів n, r і s.

Варто підкреслити, що розподіли, які використовувались для отримання випадкових задач для цих тестів, не були обрані відповідно до якої-небудь обґрунтованої методології. Дані результати приведені тут лише в якості наочної ілюстрації. Правильний спосіб вибору випадкових задач для таких тестів

є об'єктом для подальших досліджень.

У тих невдалих випадках, коли граф обходу розбиттів складається з кількох компонент зв'язності, кількість розбиттів, які нам доведеться перевірити перед тим, як ми знайдемо правильне, значною мірою залежить від розмірів тієї компоненти зв'язності, до якої належить шукане розбиття. Це є наслідком з того, що якщо ми натрапили на цикл, тоді ми маємо обрати наступне розбиття для розгляду випадковим чином, і тоді ймовірність продовжити пошук із правильного компонента зв'язності дорівнює розміру цього компонента, поділеного на розмір множини розбиттів, з якої ми обираєм. На рис. 3.5 зображено крайній випадок пошукового графа, у якому компонент зв'язності із розв'язком складається лише з 2-х вершин.

Також, на рис. 3.1 та 3.2 зображено приклади пошукових графів з лише однією компонентою зв'язності. Ці графи відповідають задачам із тривіальним ($I_0 = \overline{1..sr}$, $I_{\min} = \varnothing$, $I_{\max} = \varnothing$) і нетривіальним (іншим) розбиттям розв'язку відповідно.

На рис. 3.3 зображено приклад із двома компонентами зв'язності, а на рис. 3.4 - приклад із трьома .

На цих рисунках розбиття позначені за допомогою послідовностей символів `0`, `-` і `+`. Кожен наступний символ у послідовності позначає, що відповідний індекс належить до I_0, $I_{\min}$ чи $I_{\max}$. Розбиття, що відповідають розв'язку, позначені зеленим кольором. Їх також можна впізнати за ребром-петлею на них. Тривіальні розбиття позначені синім кольором, якщо вони не є одночасно і розбиттям розв'язку. Пошукові цикли позначені червоним. Джерельні вершини графів, що ведуть у одні й ті самі вершини, об'єднані у вершини синього кольору, на яких підписано кількість об'єднаних таким чином вершин.

У якості альтернативи до детектування циклів, ми можемо натомість детектувати повторне відвідання одних і тих самих вершини графа. Це дозволить гарантувати знаходження розв'язку за скінченний час, а також пришвидшить пошук. Проте, це потребує суттєвих затрат машинної пам'яті.

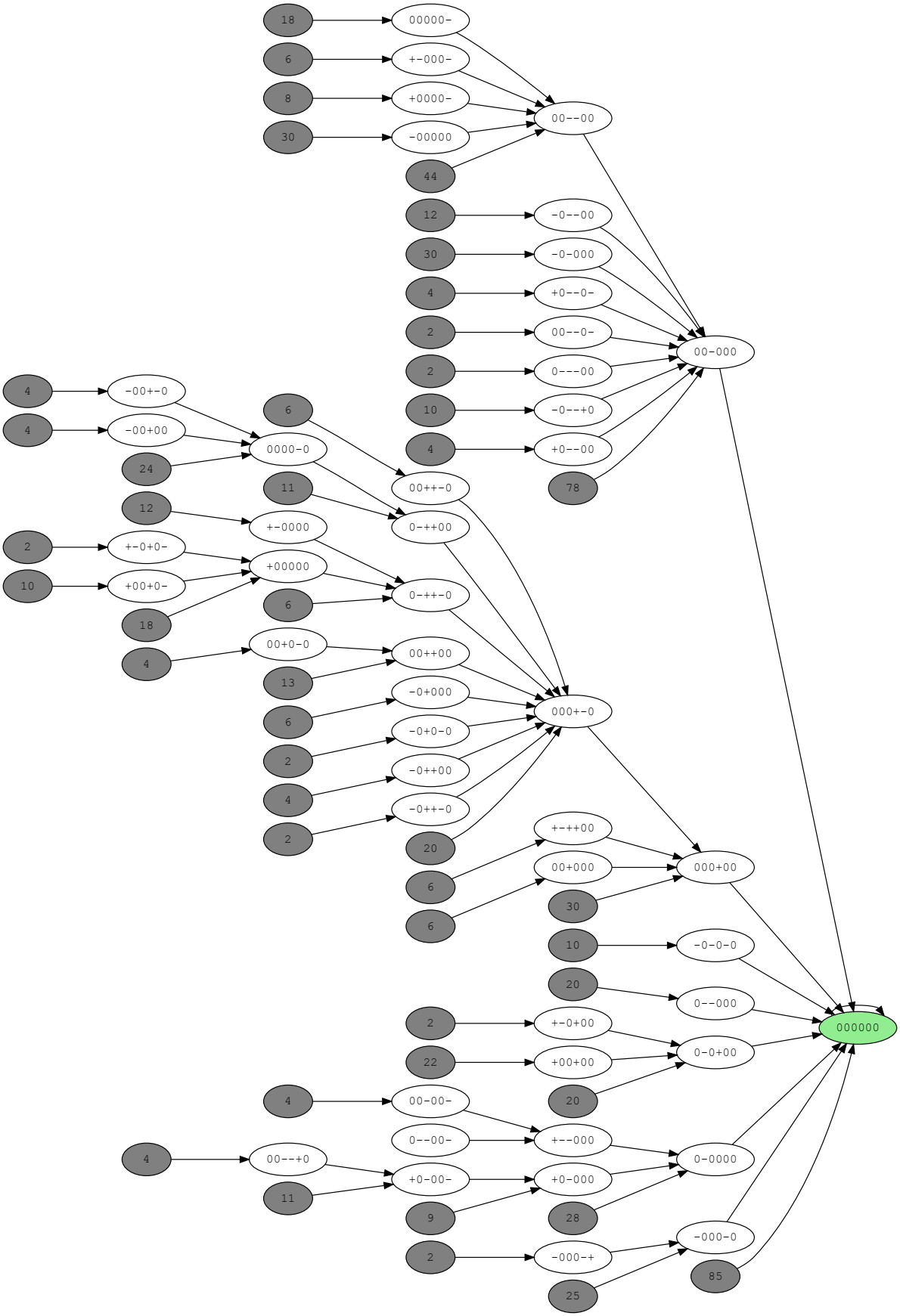

Рисунок 3.1 — Граф обходу розбиттів з одним компонентом зв'язності і тривіальним розбиттям розв'язку.

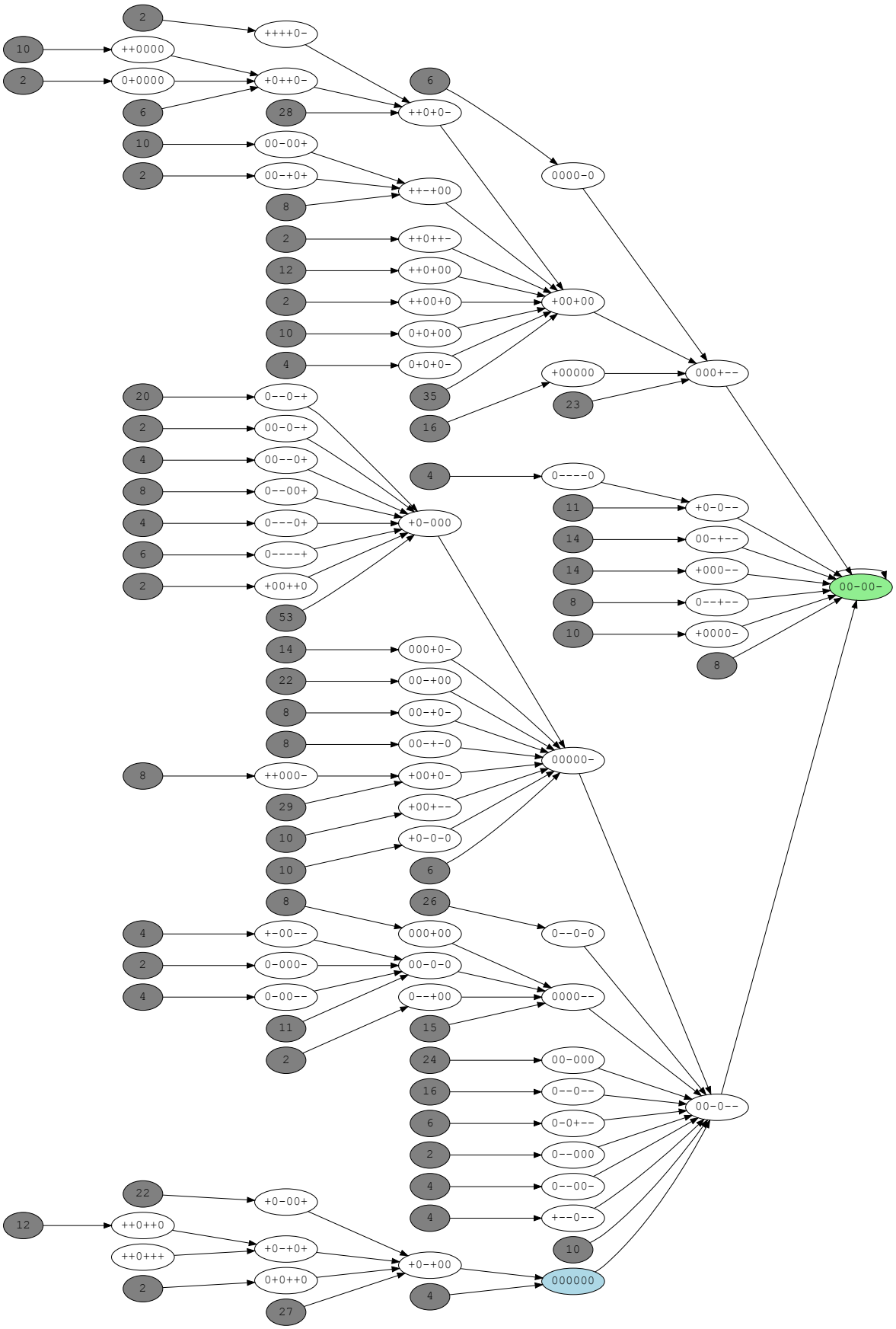

Рисунок 3.2 — Граф обходу розбиттів з одним компонентом зв'язності і нетривіальним розбиттям розв'язку.

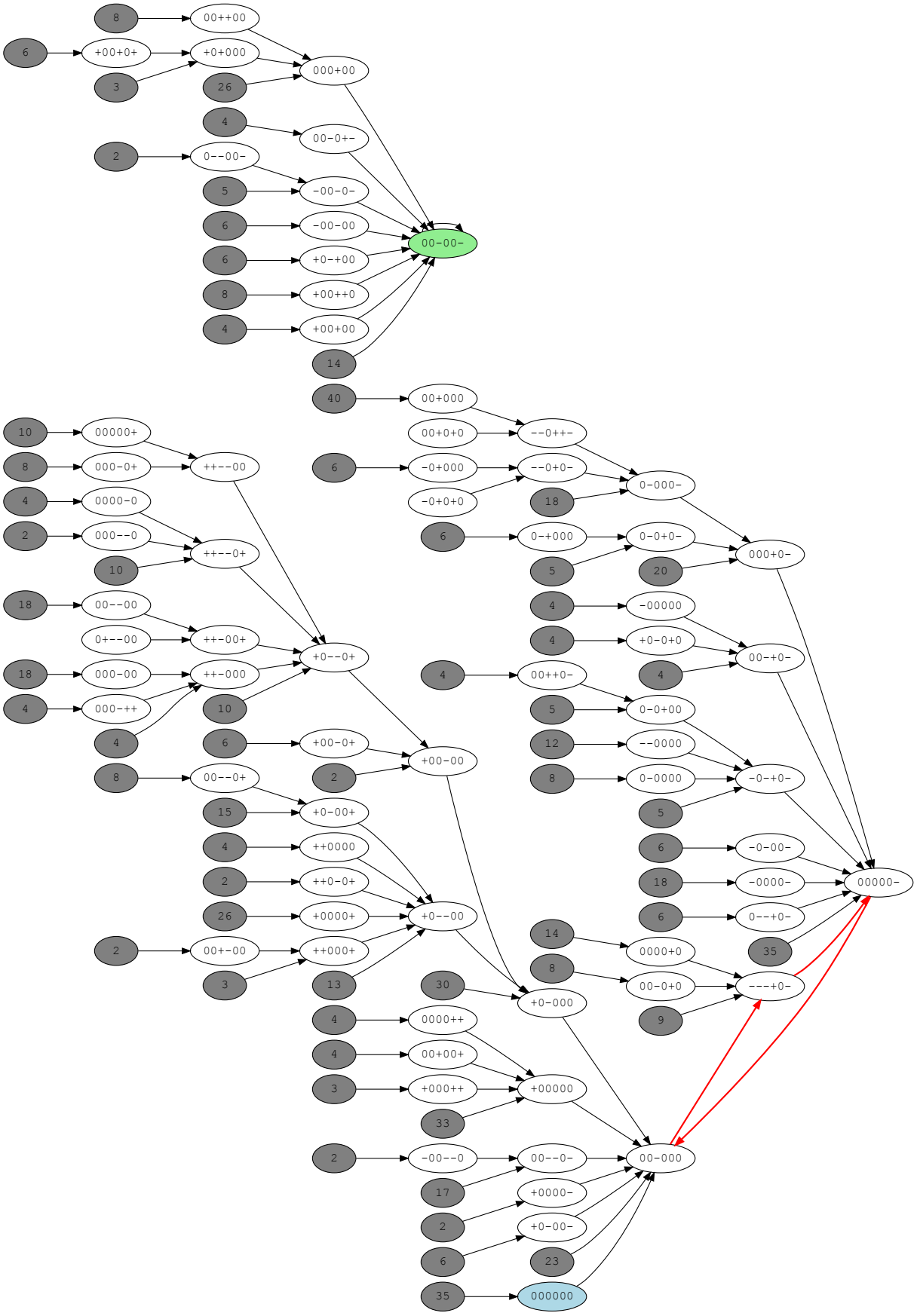

Рисунок 3.3 — Граф обходу розбиттів з двома компонентами зв'язності.

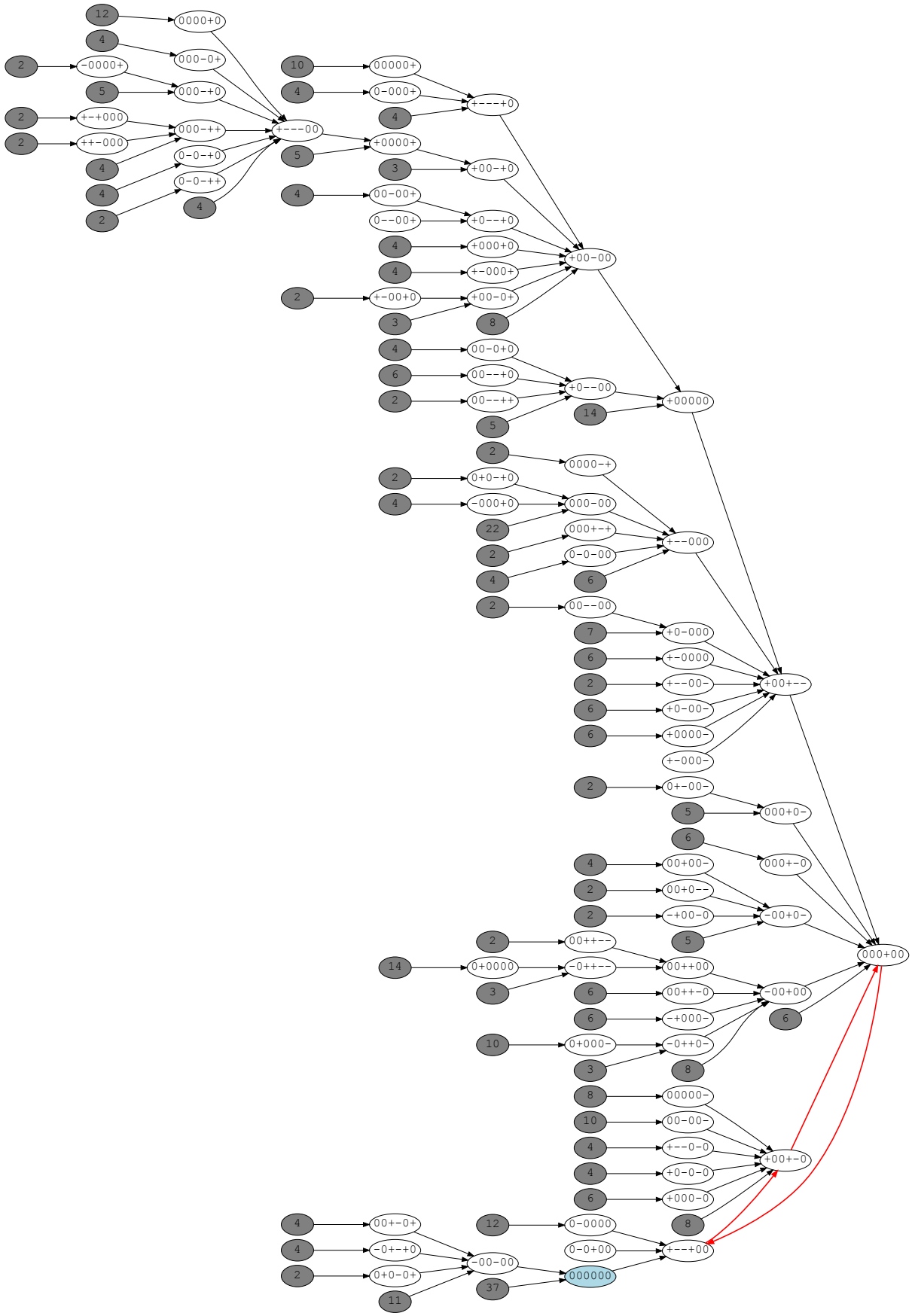

Рисунок 3.4, аркуш 1 — Граф обходу розбиттів з трьома компонентами зв'язності.

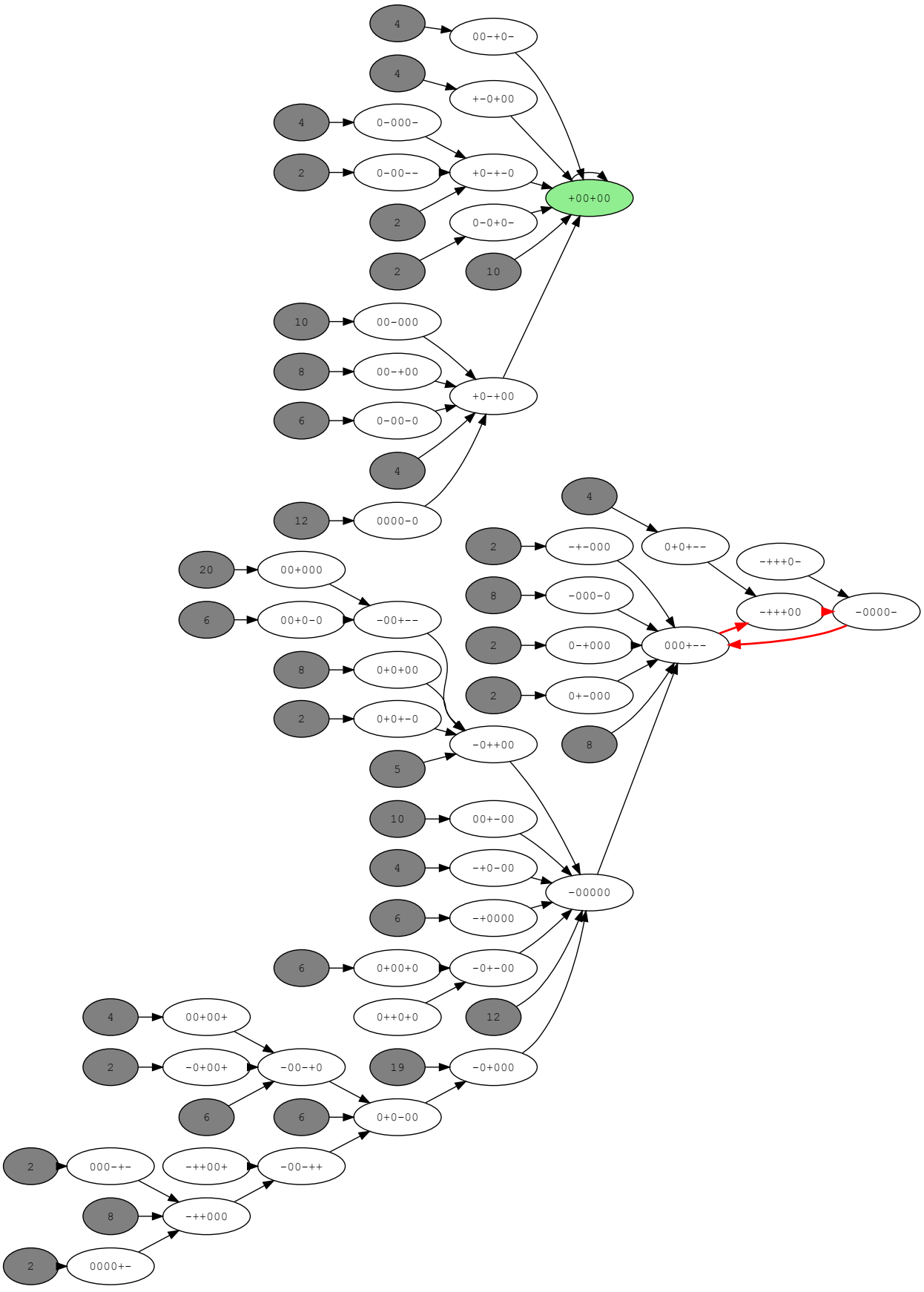

Рисунок 3.4, аркуш 2

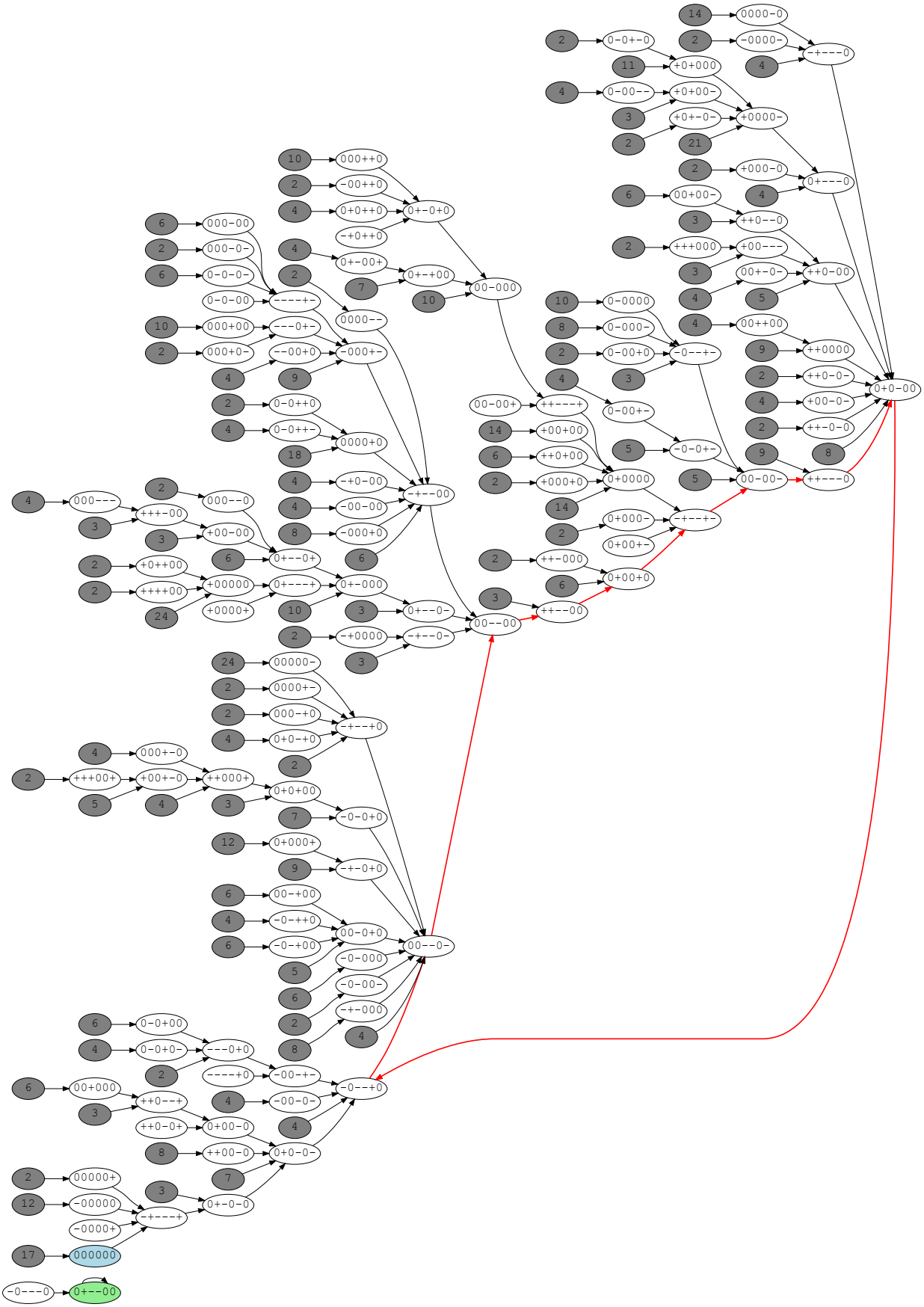

Рисунок 3.5 — Граф обходу розбиттів, у якому компонента зв'язності розв'язку складається лише з двох розбиттів.

3.4 Метод послідовних оптимумів

Даний метод народився як спроба подолати один із недоліків методу найкращої цілі - неточність, що вноситься стабілізуючим доданком $\alpha \langle \boldsymbol{u}(k,s), \boldsymbol{u}(k,s) \rangle$ функції корисності (3.54).

3.4.1 Ідея методу

Згідно проміжним результатам доведення для методу найкращої цілі, наведеному у [2, розділ 5], розв'язком задачі (3.54) при обмеженнях (1.5) є

$$\begin{cases} (\Omega_s^T \Omega_s + \alpha E)\boldsymbol{u}(k,s) = x_s^{**} + \boldsymbol{v}^{(-)*} - \boldsymbol{v}^{(+)*} \\ v_j^{(-)*}(-u_j^*(k,s) - u_{0j}) = 0, & j = \overline{1..sr} \\ v_j^{(+)*}(u_j^*(k,s) - u_{0j}) = 0, & j = \overline{1..sr} \\ v_j^{(-)*} \geqslant 0, & j = \overline{1..sr} \\ v_j^{(+)*} \geqslant 0, & j = \overline{1..sr} \\ -u_{0j} \leqslant u_j^*(k,s) \leqslant u_{0j}, & j = \overline{1..sr} \end{cases} . \tag{3.63}$$

Відомо, що ненульовий коефіцієнт α призводить до похибки. Нульове ж його значення гарантує, що ця система дає багатозначний розв'язок, і зводить функцію корисності до вигляду

$$\min \langle \Omega_s \boldsymbol{u}(k,s) - x_s^*, \Omega_s \boldsymbol{u}(k,s) - x_s^* \rangle . \tag{3.64}$$

Пропонується замість внесення додаткової похибки задля забезпечення єдиності розв'язку розв'язати додаткову оптимізаційну задачу

$$\min J_2(\boldsymbol{u}^*(k,s)) = \frac{1}{2} \langle C\boldsymbol{u}^*(k,s), \boldsymbol{u}^*(k,s) \rangle + \langle \boldsymbol{b}, \boldsymbol{u}^*(k,s) \rangle \tag{3.65}$$

на множині, заданій (3.63) за умови $\alpha = 0$. Тут матриця C - симетрична ($C^T = C$) і додатно-визначена.

3.4.2 Розбиття на підмножини

Розіб'ємо множину оптимальних за першим критерієм значень $\boldsymbol{u}^*(k,s)$, $\boldsymbol{v}^{(-)*}$ та $\boldsymbol{v}^{(+)*}$ (3.63) на підмножини, на яких можна було б застосувати теорему Куна-Такера, та тривіальні (що складаються з однієї точки або порожні) наступним чином.

Нехай множини I_0, $I_{\min}$, $I_{\max}$ такі, що виконуються умови

$$\begin{aligned} I_0 &\subset \overline{1..sr}, \\ I_{\min} &\subset \overline{1..sr}, \\ I_{\max} &\subset \overline{1..sr}, \\ I_0 \cup I_{\min} \cup I_{\max} &= \overline{1..sr}, \\ I_0 \cap I_{\min} &= \varnothing, \\ I_0 \cap I_{\max} &= \varnothing, \\ I_{\min} \cap I_{\max} &= \varnothing. \end{aligned} \tag{3.66}$$

Тоді задамо множину $U(I_0, I_{\min}, I_{\max})$ наступним чином:

$$U(I_0, I_{\min}, I_{\max}) = \{\boldsymbol{u}^*(k,s) | \begin{cases} \left(\boldsymbol{u}^*(k,s), \boldsymbol{v}^{(-)*}, \boldsymbol{v}^{(+)*}\right) \in (3.63) \\ v_j^{(-)*} > 0, \quad j \in I_{\min} \\ v_j^{(+)*} > 0, \quad j \in I_{\max} \\ v_j^{(-)*} = 0, \quad j \in I_0 \\ v_j^{(+)*} = 0, \quad j \in I_0 \end{cases} \} \tag{3.67}$$

Очевидно що виконуються наступні твердження:

$$\forall \left(\boldsymbol{u}^*(k,s), \boldsymbol{v}^{(-)*}, \boldsymbol{v}^{(+)*}\right) \in U(I_0, I_{\min}, I_{\max}) \quad j \in I_{\min} \to u_j^*(k,s) = -u_{0j}, \tag{3.68}$$

$$\forall \left(\boldsymbol{u}^*(k,s), \boldsymbol{v}^{(-)*}, \boldsymbol{v}^{(+)*}\right) \in U(I_0, I_{\min}, I_{\max}) \quad j \in I_{\min} \to v_j^{(+)*} = 0, \tag{3.69}$$

$$\forall \left(\boldsymbol{u}^*(k,s), \boldsymbol{v}^{(-)*}, \boldsymbol{v}^{(+)*}\right) \in U(I_0, I_{\min}, I_{\max}) \quad j \in I_{\max} \to u_j^*(k,s) = u_{0j}, \tag{3.70}$$

$$\forall \left(\boldsymbol{u}^*(k,s), \boldsymbol{v}^{(-)*}, \boldsymbol{v}^{(+)*}\right) \in U(I_0, I_{\min}, I_{\max}) \quad j \in I_{\max} \to v_j^{(-)*} = 0. \tag{3.71}$$

Тоді виконується вираз (3.72).

$$U(I_0, I_{\min}, I_{\max}) = \{\boldsymbol{u}^*(k,s) | \begin{cases} \Omega'_{I_0} \quad \boldsymbol{u}^*(k,s) = \boldsymbol{x}_s^{**}{}_{I_0} \\ \Omega'_{I_{\min}} \quad \boldsymbol{u}^*(k,s) < \boldsymbol{x}_s^{**}{}_{I_{\min}} \\ \Omega'_{I_{\max}} \quad \boldsymbol{u}^*(k,s) > \boldsymbol{x}_s^{**}{}_{I_{\max}} \\ -\boldsymbol{u}_{0I_0} \leqslant \boldsymbol{u}_{I_0}^*(k,s) \leqslant \boldsymbol{u}_{0I_0} \\ \boldsymbol{u}_{I_{\min}}^*(k,s) = - \quad u_{0I_{\min}} \\ \boldsymbol{u}_{I_{\max}}^*(k,s) = \quad u_{0I_{\max}} \end{cases} \}, \tag{3.72}$$

де $\Omega' = \Omega_s^T \Omega_s$.

Очевидно, що виконується також вираз

$$\bigcup_{(I_0, I_{\min}, I_{\max}) \in (3.66)} U(I_0, I_{\min}, I_{\max}) = \{\boldsymbol{u}^*(k,s) | \left(\boldsymbol{u}^*(k,s), \boldsymbol{v}^{(-)*}, \boldsymbol{v}^{(+)*}\right) \in (3.63)\}. \tag{3.73}$$

Тоді щоби розв'язати задачу (3.65) на (3.63), достатньо розв'язати її на всіх її підмножинах $U(I_0, I_{\min}, I_{\max})$ і знайти ту підмножину, на якій розв'язок даватиме найменьше значення (3.65).

3.4.3 Мінімуми на підмножинах

Запишемо (3.65) з урахуванням відомих для (3.72) значень $\boldsymbol{u}^*(k, s)$ у наступному вигляді:

$$
\begin{aligned}
J_2(\boldsymbol{u}^*(k,s)) =& \frac{1}{2}\left\langle C\boldsymbol{u}^*(k,s), \boldsymbol{u}^*(k,s)\right\rangle + \left\langle \boldsymbol{b}, \boldsymbol{u}^*(k,s)\right\rangle = \\
=& \frac{1}{2}\Bigg(\left\langle C_{I_0}\boldsymbol{u}^*(k,s), \boldsymbol{u}^*_{I_0}(k,s)\right\rangle + \left\langle C_{I_{\min}}\boldsymbol{u}^*(k,s), \boldsymbol{u}^*_{I_{\min}}(k,s)\right\rangle + \\
&+ \left\langle C_{I_{\max}}\boldsymbol{u}^*(k,s), \boldsymbol{u}^*_{I_{\max}}(k,s)\right\rangle\Bigg) + \\
&+ \left\langle \boldsymbol{b}_{I_0}, \boldsymbol{u}^*_{I_0}(k,s)\right\rangle + \left\langle \boldsymbol{b}_{I_{\min}}, \boldsymbol{u}^*_{I_{\min}}(k,s)\right\rangle + \left\langle \boldsymbol{b}_{I_{\max}}, \boldsymbol{u}^*_{I_{\max}}(k,s)\right\rangle = \\
=& \frac{1}{2}\Bigg(\left\langle C^{I_0}_{I_0}\boldsymbol{u}^*_{I_0}(k,s), \boldsymbol{u}^*_{I_0}(k,s)\right\rangle + \\
&+ \left\langle C^{I_{\min}}_{I_0}\boldsymbol{u}^*_{I_{\min}}(k,s), \boldsymbol{u}^*_{I_0}(k,s)\right\rangle + \\
&+ \left\langle C^{I_{\max}}_{I_0}\boldsymbol{u}^*_{I_{\max}}(k,s), \boldsymbol{u}^*_{I_0}(k,s)\right\rangle + \\
&+ \left\langle C^{I_0}_{I_{\min}}\boldsymbol{u}^*_{I_0}(k,s), \boldsymbol{u}^*_{I_{\min}}(k,s)\right\rangle + \\
&+ \left\langle C^{I_{\min}}_{I_{\min}}\boldsymbol{u}^*_{I_{\min}}(k,s), \boldsymbol{u}^*_{I_{\min}}(k,s)\right\rangle + \\
&+ \left\langle C^{I_{\max}}_{I_{\min}}\boldsymbol{u}^*_{I_{\max}}(k,s), \boldsymbol{u}^*_{I_{\min}}(k,s)\right\rangle + \\
&+ \left\langle C^{I_0}_{I_{\max}}\boldsymbol{u}^*_{I_0}(k,s), \boldsymbol{u}^*_{I_{\max}}(k,s)\right\rangle + \\
&+ \left\langle C^{I_{\min}}_{I_{\max}}\boldsymbol{u}^*_{I_{\min}}(k,s), \boldsymbol{u}^*_{I_{\max}}(k,s)\right\rangle + \\
&+ \left\langle C^{I_{\max}}_{I_{\max}}\boldsymbol{u}^*_{I_{\max}}(k,s), \boldsymbol{u}^*_{I_{\max}}(k,s)\right\rangle\Bigg) + \\
&+ \left\langle \boldsymbol{b}_{I_0}, \boldsymbol{u}^*_{I_0}(k,s)\right\rangle + \left\langle \boldsymbol{b}_{I_{\min}}, \boldsymbol{u}^*_{I_{\min}}(k,s)\right\rangle + \left\langle \boldsymbol{b}_{I_{\max}}, \boldsymbol{u}^*_{I_{\max}}(k,s)\right\rangle = \\
=& \frac{1}{2}\left\langle C^{I_0}_{I_0}\boldsymbol{u}^*_{I_0}(k,s), \boldsymbol{u}^*_{I_0}(k,s)\right\rangle + \\
&+ \left\langle C^{I_{min}}_{I_0}\boldsymbol{u}^*_{I_{min}}(k,s) + C^{I_{max}}_{I_0}\boldsymbol{u}^*_{I_{max}}(k,s) + \boldsymbol{b}_{I_0}, \boldsymbol{u}^*_{I_0}(k,s)\right\rangle + \\
&+ \frac{1}{2}\left\langle C^{I_{\min}}_{I_{\min}}\boldsymbol{u}^*_{I_{\min}}(k,s), \boldsymbol{u}^*_{I_{\min}}(k,s)\right\rangle + \\
&+ \frac{1}{2}\left\langle C^{I_{\max}}_{I_{\max}}\boldsymbol{u}^*_{I_{\max}}(k,s), \boldsymbol{u}^*_{I_{\max}}(k,s)\right\rangle + \\
&+ \left\langle C^{I_{\max}}_{I_{\min}}\boldsymbol{u}^*_{I_{\max}}(k,s), \boldsymbol{u}^*_{I_{\min}}(k,s)\right\rangle + \\
&+ \left\langle \boldsymbol{b}_{I_{\min}}, \boldsymbol{u}^*_{I_{\min}}(k,s)\right\rangle + \left\langle \boldsymbol{b}_{I_{\max}}, \boldsymbol{u}^*_{I_{\max}}(k,s)\right\rangle
\end{aligned}
\tag{3.74}
$$

Зверну увагу на той факт, що, якщо врахувати, що $\boldsymbol{u}^*_{I_{min}}(k,s)$ і $\boldsymbol{u}^*_{I_{max}}(k,s)$ відомі, то (3.74) має вигляд, аналогічний (3.65) плюс константа.

Перед застосуванні теореми Куна-Такера необхідно також зазначити, що у формулюванні тереми застосовуються нестрогі нерівності, тоді як у (3.72) використовуються строгі нерівності. Тому далі працюватимемо із замиканням цієї множини ($\overline{U(I_0, I_{\min}, I_{\max})}$). Легко бачити, що множина, задана (3.63), є замкнутою. Тому раз множина $U(I_0, I_{\min}, I_{\max})$ є її підмножиною, то і її замикання $\overline{U(I_0, I_{\min}, I_{\max})}$ також є її підмножиною.

Також легко бачити, що $\overline{U(I_0, I_{\min}, I_{\max})}$ можна записати у вигляді

$$\overline{U(I_0, I_{\min}, I_{\max})} = \{\boldsymbol{u}^*(k,s) | \begin{cases} \Omega'_{I_0} \quad \boldsymbol{u}^*(k,s) = \boldsymbol{x}^{**}_{s\ I_0} \\ \Omega'_{I_{\min}} \quad \boldsymbol{u}^*(k,s) \leqslant \boldsymbol{x}^{**}_{s\ I_{\min}} \\ \Omega'_{I_{\max}} \quad \boldsymbol{u}^*(k,s) \geqslant \boldsymbol{x}^{**}_{s\ I_{\max}} \\ -\boldsymbol{u}_{0I_0} \leqslant \boldsymbol{u}^*_{I_0}(k,s) \leqslant \boldsymbol{u}_{0I_0} \\ u^*_{I_{\min}}(k,s) = - \quad u_{0I_{\min}} \\ u^*_{I_{\max}}(k,s) = \quad u_{0I_{\max}} \end{cases} \}. \tag{3.75}$$

Застосуємо теорему Куна-Такера на $\overline{U(I_0, I_{\min}, I_{\max})}$ із цільовою функцією (3.74). Функція Лагранжа тоді прийматиме вигляд

$$\begin{aligned} \Lambda_{2(I_0,I_{\min},I_{\max})}(\boldsymbol{u}^*_{I_0}(k,s), \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\lambda}) = \frac{1}{2}\left\langle C^{I_0}_{I_0}\boldsymbol{u}^*_{I_0}(k,s), \boldsymbol{u}^*_{I_0}(k,s)\right\rangle + \\ + \left\langle C^{I_{min}}_{I_0}\boldsymbol{u}^*_{I_{min}}(k,s) + C^{I_{max}}_{I_0}\boldsymbol{u}^*_{I_{max}}(k,s) + \boldsymbol{b}_{I_0}, \boldsymbol{u}^*_{I_0}(k,s)\right\rangle + \text{const} + \\ + \left\langle \boldsymbol{\lambda}, \Omega'^{I_0}_{I_0}\boldsymbol{u}^*_{I_0}(k,s) + \Omega'^{I_{\min}}_{I_0}\boldsymbol{u}^*_{I_{\min}}(k,s) + \Omega'^{I_{\max}}_{I_0}\boldsymbol{u}^*_{I_{\max}}(k,s) - \boldsymbol{x}^{**}_{s\ I_0}\right\rangle + \\ + \left\langle \boldsymbol{\alpha}, \Omega'^{I_0}_{I_{\min}}\boldsymbol{u}^*_{I_0}(k,s) + \Omega'^{I_{\min}}_{I_{\min}}\boldsymbol{u}^*_{I_{\min}}(k,s) + \Omega'^{I_{\max}}_{I_{\min}}\boldsymbol{u}^*_{I_{\max}}(k,s) - \boldsymbol{x}^{**}_{s\ I_{\min}}\right\rangle - \\ - \left\langle \boldsymbol{\beta}, \Omega'^{I_0}_{I_{\max}}\boldsymbol{u}^*_{I_0}(k,s) + \Omega'^{I_{\min}}_{I_{\max}}\boldsymbol{u}^*_{I_{\min}}(k,s) + \Omega'^{I_{\max}}_{I_{\max}}\boldsymbol{u}^*_{I_{\max}}(k,s) - \boldsymbol{x}^{**}_{s\ I_{\max}}\right\rangle + \\ + \left\langle \boldsymbol{\gamma}, \boldsymbol{u}^*_{I_0}(k,s) - \boldsymbol{u}_{0I_0}\right\rangle + \\ + \left\langle \boldsymbol{\epsilon}, -\boldsymbol{u}^*_{I_0}(k,s) - \boldsymbol{u}_{0I_0}\right\rangle \end{aligned} \tag{3.76}$$

і буде визначена на області

$$\begin{cases} \boldsymbol{u}^*(k,s) \in \overline{U(I_0, I_{\min}, I_{\max})} \\ \boldsymbol{\alpha} \geqslant 0 \\ \boldsymbol{\beta} \geqslant 0 \\ \boldsymbol{\gamma} \geqslant 0 \\ \boldsymbol{\epsilon} \geqslant 0 \end{cases} . \tag{3.77}$$

Її часткова похідна матиме вигляд

$$\frac{\partial \Lambda_{2(I_0,I_{\min},I_{\max})}}{\partial \boldsymbol{u}^*_{I_0}(k,s)} = C^{I_0}_{I_0}\boldsymbol{u}^*_{I_0}(k,s) + \\ + \left(\boldsymbol{b}_{I_0} + C^{I_{\max}}_{I_0}\boldsymbol{u}^*_{I_{\max}}(k,s) + C^{I_{\min}}_{I_0}\boldsymbol{u}^*_{I_{\min}}(k,s)\right) + \\ + \Omega'^{I_0}_{I_0}\boldsymbol{\lambda} + \Omega'^{I_{\min}}_{I_0}\boldsymbol{\alpha} + \Omega'^{I_{\max}}_{I_0}\boldsymbol{\beta} + \boldsymbol{\gamma} - \boldsymbol{\epsilon}. \tag{3.78}$$

Запишемо умови доповнюючої нев'язки:

$$\alpha_i \Bigg(\Omega'^{I_0}_{I_{\min}[i]}\boldsymbol{u}^*_{I_0}(k,s) + \Omega'^{I_{\max}}_{I_{\min}[i]}\boldsymbol{u}^*_{I_{\max}}(k,s) + \\ + \Omega'^{I_{\min}}_{I_{\min}[i]}\boldsymbol{u}^*_{I_{\min}}(k,s) - \boldsymbol{x}^{**}_{s\,I_{\min}[i]} \Bigg) = 0, \quad i \in \overline{1..\operatorname{card} I_{\min}}, \tag{3.79}$$

$$\beta_i \Bigg(\Omega'^{I_0}_{I_{\max}[i]}\boldsymbol{u}^*_{I_0}(k,s) + \Omega'^{I_{\max}}_{I_{\max}[i]}\boldsymbol{u}^*_{I_{\max}}(k,s) + \\ + \Omega'^{I_{\min}}_{I_{\max}[i]}\boldsymbol{u}^*_{I_{\min}}(k,s) - \boldsymbol{x}^{**}_{s\,I_{\max}[i]} \Bigg) = 0, \quad i \in \overline{1..\operatorname{card} I_{\max}}, \tag{3.80}$$

$$\gamma_i \left(\boldsymbol{u}^*_{I_0[i]}(k,s) - \boldsymbol{u}_{0I_0[i]} \right) = 0, \qquad i \in \overline{1..\operatorname{card} I_0}, \tag{3.81}$$

$$\epsilon_i \left(-\boldsymbol{u}^*_{I_0[i]}(k,s) - \boldsymbol{u}_{0I_0[i]} \right) = 0, \qquad i \in \overline{1..\operatorname{card} I_0}. \tag{3.82}$$

Таким чином, мінімум функціоналу (3.74) на множині $\overline{U(I_0, I_{\min}, I_{\max})}$ доставляє система (3.77), (3.79), (3.80), (3.81), (3.82) разом із виразом

$$\frac{\partial \Lambda_{2(I_0, I_{\min}, I_{\max})}}{\partial \boldsymbol{u}^*_{I_0}(k,s)} = 0. \tag{3.83}$$

***Лема* 1.** *Нехай* $(\boldsymbol{u}^*_{I_0}(k,s), \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\epsilon})$ *задовольняють* (3.83), (3.77), (3.79), (3.80), (3.81), (3.82) *і* $\gamma_i > 0$. *Тоді* $u^*_i(k,s) = u_{0i}$, $\boldsymbol{u}^*(k,s) \in \overline{U(I_0 \backslash \{I_0[i]\}, I_{\min}, I_{\max} \cup \{I_0[i]\})}$ *і* $\boldsymbol{u}^*(k,s)$ *доставляє мінімум функціоналу* J_2 *на* $\overline{U(I_0 \backslash \{I_0[i]\}, I_{\min}, I_{\max} \cup \{I_0[i]\})}$

***Лема* 2.** *Нехай* $(\boldsymbol{u}^*_{I_0}(k,s), \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\epsilon})$ *задовольняють* (3.83), (3.77), (3.79), (3.80), (3.81), (3.82) *і* $\epsilon_i > 0$. *Тоді* $u^*_i(k,s) = -u_{0i}$, $\boldsymbol{u}^*(k,s) \in \overline{U(I_0 \backslash \{I_0[i]\}, I_{\min} \cup \{I_0[i]\}, I_{\max})}$ *і* $\boldsymbol{u}^*(k,s)$ *доставляє мінімум функціоналу* J_2 *на* $\overline{U(I_0 \backslash \{I_0[i]\}, I_{\min} \cup \{I_0[i]\}, I_{\max})}$

***Лема* 3.** *Нехай* $(\boldsymbol{u}^*_{I_0}(k,s), \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\epsilon})$ *задовольняють* (3.83), (3.77), (3.79), (3.80), (3.81), (3.82) *і* $\alpha_i > 0$. *Тоді* $\boldsymbol{u}^*(k,s) \in \overline{U(I_0 \cup \{I_{\min}[i]\}, I_{\min} \backslash \{I_{\min}[i]\}, I_{\max})}$ *і* $\boldsymbol{u}^*(k,s)$ *доставляє мінімум функціоналу* J_2 *на* $\overline{U(I_0 \cup \{I_{\min}[i]\}, I_{\min} \backslash \{I_{\min}[i]\}, I_{\max})}$.

***Лема* 4.** *Нехай* $(\boldsymbol{u}^*_{I_0}(k,s), \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\gamma}, \boldsymbol{\epsilon})$ *задовольняють* (3.83), (3.77), (3.79), (3.80), (3.81), (3.82) *і* $\beta_i > 0$. *Тоді* $\boldsymbol{u}^*(k,s) \in \overline{U(I_0 \cup \{I_{\max}[i]\}, I_{\min}, I_{\max} \backslash \{I_{\max}[i]\})}$ *і* $\boldsymbol{u}^*(k,s)$ *доставляє мінімум функціоналу* J_2 *на* $\overline{U(I_0 \cup \{I_{\max}[i]\}, I_{\min}, I_{\max} \backslash \{I_{\max}[i]\})}$.

***Лема* 5** (Наслідок з лемм 1, 2, 3 і 4)**.** *Нехай* $\boldsymbol{u}^*(k,s)$ *доставляє мінімум функціоналу* J_2 *на деякій множині* $\overline{U(I_0, I_{\min}, I_{\max})}$. *Тоді* $\boldsymbol{u}^*(k,s)$ *доставляє мінімум функціоналу* J_2 *на деякій множині* $\overline{U(I'_0, I'_{\min}, I'_{\max})}$ *такій, що у системі* (3.83), (3.77), (3.79), (3.80), (3.81), (3.82) *записаній для цієї множини* $\boldsymbol{\alpha} = 0$, $\boldsymbol{\beta} = 0$, $\boldsymbol{\gamma} = 0$, $\boldsymbol{\epsilon} = 0$.

Таким чином, глобальний мінімум J_2 на (3.63) слід шукати серед розв'язків систем рівнянь

$$C_{I_0}^{I_0}\boldsymbol{u}_{I_0}^{**}(k,s)+\left(\boldsymbol{b}_{I_0}+C_{I_0}^{I_{\max}}\boldsymbol{u}_{I_{\max}}^{*}(k,s)+C_{I_0}^{I_{\min}}\boldsymbol{u}_{I_{\min}}^{*}(k,s)\right)+\Omega'^{I_0}_{I_0}\boldsymbol{\lambda}=0 \quad (3.84)$$

на (3.75), записаних для всіх можливих I_0, $I_{\min}$, $I_{\max}$, які задовольняють (3.66). Тут і далі $\boldsymbol{u}_{I_0}^{**}(k,s)$ - значення керування, оптимальне спочатку по J_1, а потім по J_2.

3.4.4 Обчислення мінімуму на підмножині

В подальшому нам знадобляться дві наступні леми: лема 6 і 7.

Лема 6. *Нехай* $C>0$. *Тоді* $\forall I\subset\overline{1..\dim C}\ C_I^I>0$.

Лема 7. *Нехай* $C>0$. *Тоді існує* C^{-1}.

Із (3.84) отримуємо

$$\boldsymbol{u}_{I_0}^{**}(k,s)=-{C_{I_0}^{I_0}}^{-1}\left(\boldsymbol{b}_{I_0}+C_{I_0}^{I_{\max}}\boldsymbol{u}_{I_{\max}}^{*}(k,s)+C_{I_0}^{I_{\min}}\boldsymbol{u}_{I_{\min}}^{*}(k,s)\right)- \\ -{C_{I_0}^{I_0}}^{-1}\Omega'^{I_0}_{I_0}\boldsymbol{\lambda}. \quad (3.85)$$

Таким чином для знаходження $\boldsymbol{u}_{I_0}^{**}(k,s)$ для заданих I_0, $I_{\min}$, $I_{\max}$ достатньо знайти значення виразу ${C_{I_0}^{I_0}}^{-1}\Omega'^{I_0}_{I_0}\boldsymbol{\lambda}$.

Підставимо (3.85) у рівність із формального запису $\overline{U(I_0,I_{\min},I_{\max})}$ (3.75). Таким чином отримуємо системи рівнянь

$$-\Omega'^{I_0}_{I_0}{C_{I_0}^{I_0}}^{-1}\left(\boldsymbol{b}_{I_0}+C_{I_0}^{I_{\max}}\boldsymbol{u}_{I_{\max}}^{*}(k,s)+C_{I_0}^{I_{\min}}\boldsymbol{u}_{I_{\min}}^{*}(k,s)\right)- \\ -\Omega'^{I_0}_{I_0}{C_{I_0}^{I_0}}^{-1}\Omega'^{I_0}_{I_0}\boldsymbol{\lambda}+ \\ +\Omega'^{I_{\min}}_{I_0}\boldsymbol{u}_{I_{\min}}^{*}(k,s)+ \\ +\Omega'^{I_{\max}}_{I_0}\boldsymbol{u}_{I_{\max}}^{*}(k,s)=\boldsymbol{x}_s^{**}{}_{I_0} \quad \text{і} \quad (3.86)$$

$$\Omega'^{I_0}_{I_0} {C^{I_0}_{I_0}}^{-1} \Omega'^{I_0}_{I_0} \boldsymbol{\lambda} = -{\boldsymbol{x}^{**}_s}_{I_0} - \\ - \Omega'^{I_0}_{I_0} {C^{I_0}_{I_0}}^{-1} \left(\boldsymbol{b}_{I_0} + C^{I_{\max}}_{I_0} \boldsymbol{u}^*_{I_{\max}}(k,s) + C^{I_{\min}}_{I_0} \boldsymbol{u}^*_{I_{\min}}(k,s) \right) + \\ + \Omega'^{I_{\min}}_{I_0} \boldsymbol{u}^*_{I_{\min}}(k,s) + \\ + \Omega'^{I_{\max}}_{I_0} \boldsymbol{u}^*_{I_{\max}}(k,s). \quad (3.87)$$

***Лема* 8.** *Нехай* $A = A^T$. *Тоді* $\operatorname{Im} A \perp \operatorname{Ker} A$.

Доведення. Нехай $\boldsymbol{u} \in \operatorname{Im} A$ і $\boldsymbol{v} \in \operatorname{Ker} A$. Тоді $\exists \boldsymbol{u}' : \ \boldsymbol{u} = A\boldsymbol{u}'$.

$$\langle \boldsymbol{u}, \boldsymbol{v} \rangle = \langle A\boldsymbol{u}', \boldsymbol{v} \rangle = \left\langle \boldsymbol{u}', A^T \boldsymbol{v} \right\rangle = \langle \boldsymbol{u}', A\boldsymbol{v} \rangle = \langle \boldsymbol{u}', \boldsymbol{0} \rangle = 0.$$

□

***Лема* 9.** *Якщо для деякого* $\boldsymbol{b}$ $\exists \boldsymbol{\lambda} : \Omega'^{I_0}_{I_0} {C^{I_0}_{I_0}}^{-1} \Omega'^{I_0}_{I_0} \boldsymbol{\lambda} = \boldsymbol{b}$, *тоді* $\exists! \boldsymbol{\gamma} \in \operatorname{Im} {C^{I_0}_{I_0}}^{-1} \Omega'^{I_0}_{I_0} : \Omega'^{I_0}_{I_0} \boldsymbol{\gamma} = \boldsymbol{b}$.

Доведення. Припустімо, що

$$\exists \boldsymbol{\gamma}_1, \boldsymbol{\gamma}_2 \in \operatorname{Im} {C^{I_0}_{I_0}}^{-1} \Omega'^{I_0}_{I_0} : \ \boldsymbol{\gamma}_1 \neq \boldsymbol{\gamma}_2, \ \Omega'^{I_0}_{I_0} \boldsymbol{\gamma}_1 = \Omega'^{I_0}_{I_0} \boldsymbol{\gamma}_1.$$

Тоді $\boldsymbol{\gamma}_1 - \boldsymbol{\gamma}_2 \in \operatorname{Ker} \Omega'^{I_0}_{I_0}$.

$$\exists \boldsymbol{\lambda}_1 : {C^{I_0}_{I_0}}^{-1} \Omega'^{I_0}_{I_0} \boldsymbol{\lambda}_1 = \boldsymbol{\gamma}_1$$
$$\exists \boldsymbol{\lambda}_2 : {C^{I_0}_{I_0}}^{-1} \Omega'^{I_0}_{I_0} \boldsymbol{\lambda}_2 = \boldsymbol{\gamma}_2$$

$$\Omega'^{I_0}_{I_0} \boldsymbol{\lambda}_1 \in \operatorname{Im} \Omega'^{I_0}_{I_0}$$
$$\Omega'^{I_0}_{I_0} \boldsymbol{\lambda}_2 \in \operatorname{Im} \Omega'^{I_0}_{I_0}$$
$$\Omega'^{I_0}_{I_0} \boldsymbol{\lambda}_1 - \Omega'^{I_0}_{I_0} \boldsymbol{\lambda}_2 \in \operatorname{Im} \Omega'^{I_0}_{I_0}$$

${\Omega'}_{I_0}^{I_0} = {\Omega'}_{I_0}^{I_0 T}$, отже за лемою 8 $\operatorname{Im} {\Omega'}_{I_0}^{I_0} \perp \operatorname{Ker} {\Omega'}_{I_0}^{I_0}$.

$$0 = \left\langle \boldsymbol{\gamma}_1 - \boldsymbol{\gamma}_2, \left({\Omega'}_{I_0}^{I_0} \boldsymbol{\lambda}_1 - {\Omega'}_{I_0}^{I_0} \boldsymbol{\lambda}_2 \right) \right\rangle = \left\langle {C_{I_0}^{I_0}}^{-1} \left({\Omega'}_{I_0}^{I_0} \boldsymbol{\lambda}_1 - {\Omega'}_{I_0}^{I_0} \boldsymbol{\lambda}_2 \right), \left({\Omega'}_{I_0}^{I_0} \boldsymbol{\lambda}_1 - {\Omega'}_{I_0}^{I_0} \boldsymbol{\lambda}_2 \right) \right\rangle$$

Але ${C_{I_0}^{I_0}}^{-1}$ - додатньо-визначена. Тому

$$ {\Omega'}_{I_0}^{I_0} \boldsymbol{\lambda}_1 - {\Omega'}_{I_0}^{I_0} \boldsymbol{\lambda}_2 = \mathbf{0}. $$

Але тоді

$$ \boldsymbol{\gamma}_1 - \boldsymbol{\gamma}_2 = {C_{I_0}^{I_0}}^{-1} \left({\Omega'}_{I_0}^{I_0} \boldsymbol{\lambda}_1 - {\Omega'}_{I_0}^{I_0} \boldsymbol{\lambda}_2 \right) = {C_{I_0}^{I_0}}^{-1} \mathbf{0} = \mathbf{0}. $$

Протиріччя. □

***Лема* 10.** *Якщо для деякого* $\boldsymbol{b}$ $\exists \boldsymbol{\lambda} : {\Omega'}_{I_0}^{I_0} {C_{I_0}^{I_0}}^{-1} {\Omega'}_{I_0}^{I_0} \boldsymbol{\lambda} = \boldsymbol{b}$ *і* I_r *- найбільша підмножина* I_0 *така, що* ${\Omega'}_{I_r}^{I_0} {C_{I_0}^{I_0}}^{-1} {\Omega'}_{I_0}^{I_r}$ *- обортна матриця, тоді*

$$ {C_{I_0}^{I_0}}^{-1} {\Omega'}_{I_0}^{I_0} \boldsymbol{\lambda} = {C_{I_0}^{I_0}}^{-1} {\Omega'}_{I_0}^{I_r} \left({\Omega'}_{I_r}^{I_0} {C_{I_0}^{I_0}}^{-1} {\Omega'}_{I_0}^{I_r} \right)^{-1} \boldsymbol{b}_{I_r}. $$

***Лема* 11.** *Система рівнянь*

$$ {\Omega'}_{I_0}^{I_0} {C_{I_0}^{I_0}}^{-1} {\Omega'}_{I_0}^{I_0} \boldsymbol{\lambda} = \boldsymbol{b} $$

має розв'язок тоді і тільки тоді, коли $\boldsymbol{b} \in \operatorname{Im} {\Omega'}_{I_0}^{I_0}$.

Доведення. У випадку, коли $\boldsymbol{b} = \mathbf{0}$, твердження очевидне. Далі розглядатимемо випадок $\boldsymbol{b} \neq \mathbf{0}$.

Якщо система має розв'язок, тоді очевидно, що $\boldsymbol{b} \in \operatorname{Im} {\Omega'}_{I_0}^{I_0}$. Залишилось довести, що якщо $\boldsymbol{b} \in \operatorname{Im} {\Omega'}_{I_0}^{I_0}$, тоді система має розв'язок.

Нехай $\boldsymbol{b} \in \operatorname{Im} {\Omega'}_{I_0}^{I_0}$, але система не має розв'язку.

$$\exists \boldsymbol{d} \neq \mathbf{0}: \ {\Omega'^{I_0}_{I_0}}^{-1} \boldsymbol{b} = \boldsymbol{d} + \operatorname{Ker} \Omega'^{I_0}_{I_0}$$

$$C^{I_0}_{I_0} {\Omega'^{I_0}_{I_0}}^{-1} \boldsymbol{b} = C^{I_0}_{I_0} \boldsymbol{d} + C^{I_0}_{I_0} \operatorname{Ker} \Omega'^{I_0}_{I_0}$$

$$C^{I_0}_{I_0} \boldsymbol{d} + C^{I_0}_{I_0} \operatorname{Ker} \Omega'^{I_0}_{I_0} \cap \operatorname{Im} \Omega'^{I_0}_{I_0} = \varnothing$$

$$C^{I_0}_{I_0} \boldsymbol{d} \notin C^{I_0}_{I_0} \operatorname{Ker} \Omega'^{I_0}_{I_0} + \operatorname{Im} \Omega'^{I_0}_{I_0}$$

$$\dim \left(C^{I_0}_{I_0} \operatorname{Ker} \Omega'^{I_0}_{I_0} + \operatorname{Im} \Omega'^{I_0}_{I_0} \right) < \dim \Omega'^{I_0}_{I_0} = \dim \left(C^{I_0}_{I_0} \operatorname{Ker} \Omega'^{I_0}_{I_0} \right) + \dim \left(\operatorname{Im} \Omega'^{I_0}_{I_0} \right)$$

$$\exists \boldsymbol{e} \in \operatorname{Ker} \Omega'^{I_0}_{I_0}, \boldsymbol{f} \in \operatorname{Im} \Omega'^{I_0}_{I_0}: \ \boldsymbol{e} \neq \mathbf{0}, \boldsymbol{f} \neq \mathbf{0}, C^{I_0}_{I_0} \boldsymbol{e} = \boldsymbol{f}$$

$\Omega'^{I_0}_{I_0}$ - симетрична, тому за лемою 8 $\operatorname{Im} \Omega'^{I_0}_{I_0} \perp \operatorname{Ker} \Omega'^{I_0}_{I_0}$.
Тому $\mathbf{0} = \langle \boldsymbol{e}, \boldsymbol{f} \rangle = \left\langle \boldsymbol{e}, C^{I_0}_{I_0} \boldsymbol{e} \right\rangle$. Але $C^{I_0}_{I_0}$ - додатньо-визначена. Протиріччя.
□

Права частина рівняння (3.87) належить $\operatorname{Im} \Omega'^{I_0}_{I_0}$, тому, згідно леми 11, його розв'язок завжди існує, а згідно леми 10 можна записати вираз

$$\begin{aligned} {C^{I_0}_{I_0}}^{-1} \Omega'^{I_0}_{I_0} \boldsymbol{\lambda} = {C^{I_0}_{I_0}}^{-1} \Omega'^{I_r}_{I_0} \left(\Omega'^{I_0}_{I_r} {C^{I_0}_{I_0}}^{-1} \Omega'^{I_r}_{I_0} \right)^{-1} \Bigg(-{\boldsymbol{x}^{**}_s}_{I_r} - \\ - \Omega'^{I_0}_{I_r} {C^{I_0}_{I_0}}^{-1} \left(\boldsymbol{b}_{I_0} + C^{I_{\max}}_{I_0} \boldsymbol{u}^*_{I_{\max}}(k,s) + C^{I_{\min}}_{I_0} \boldsymbol{u}^*_{I_{\min}}(k,s) \right) + \\ + \Omega'^{I_{\min}}_{I_r} \boldsymbol{u}^*_{I_{\min}}(k,s) + \\ + \Omega'^{I_{\max}}_{I_r} \boldsymbol{u}^*_{I_{\max}}(k,s) \Bigg). \end{aligned}$$

Його також можна записати як

$$\begin{aligned} {C^{I_0}_{I_0}}^{-1} \Omega'^{I_0}_{I_0} \boldsymbol{\lambda} = -{C^{I_0}_{I_0}}^{-1} \Omega'^{I_r}_{I_0} \left(\Omega'^{I_0}_{I_r} {C^{I_0}_{I_0}}^{-1} \Omega'^{I_r}_{I_0} \right)^{-1} {\boldsymbol{x}^{**}_s}_{I_r} - \\ - {C^{I_0}_{I_0}}^{-1} \Omega'^{I_r}_{I_0} \left(\Omega'^{I_0}_{I_r} {C^{I_0}_{I_0}}^{-1} \Omega'^{I_r}_{I_0} \right)^{-1} \Omega'^{I_0}_{I_r} {C^{I_0}_{I_0}}^{-1} \boldsymbol{b}_{I_0} + \\ + {C^{I_0}_{I_0}}^{-1} \Omega'^{I_r}_{I_0} \left(\Omega'^{I_0}_{I_r} {C^{I_0}_{I_0}}^{-1} \Omega'^{I_r}_{I_0} \right)^{-1} \left(-\Omega'^{I_0}_{I_r} {C^{I_0}_{I_0}}^{-1} C^{I_{\max}}_{I_0} + \Omega'^{I_{\max}}_{I_r} \right) \boldsymbol{u}^*_{I_{\max}}(k,s) + \\ + {C^{I_0}_{I_0}}^{-1} \Omega'^{I_r}_{I_0} \left(\Omega'^{I_0}_{I_r} {C^{I_0}_{I_0}}^{-1} \Omega'^{I_r}_{I_0} \right)^{-1} \left(-\Omega'^{I_0}_{I_r} {C^{I_0}_{I_0}}^{-1} C^{I_{\min}}_{I_0} + \Omega'^{I_{\min}}_{I_r} \right) \boldsymbol{u}^*_{I_{\min}}(k,s). \end{aligned}$$

Таким чином, ми можемо безосередньо записати єдиний можливий розв'язок системи рівнянь (3.84) на (3.75) можна записати як

$$\begin{aligned}\boldsymbol{u}^{**}_{I_0}(k,s) = -{C^{I_0}_{I_0}}^{-1}\left(\boldsymbol{b}_{I_0} + C^{I_{\max}}_{I_0}\boldsymbol{u}^{*}_{I_{\max}}(k,s) + C^{I_{\min}}_{I_0}\boldsymbol{u}^{*}_{I_{\min}}(k,s)\right) + \\ + {C^{I_0}_{I_0}}^{-1}\Omega'^{I_r}_{I_0}\left(\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1}\Omega'^{I_r}_{I_0}\right)^{-1}{\boldsymbol{x}^{**}_{s}}_{I_r} + \\ + {C^{I_0}_{I_0}}^{-1}\Omega'^{I_r}_{I_0}\left(\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1}\Omega'^{I_r}_{I_0}\right)^{-1}\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1}\boldsymbol{b}_{I_0} - \\ - {C^{I_0}_{I_0}}^{-1}\Omega'^{I_r}_{I_0}\left(\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1}\Omega'^{I_r}_{I_0}\right)^{-1}\left(-\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1}C^{I_{\max}}_{I_0} + \Omega'^{I_{\max}}_{I_r}\right)\boldsymbol{u}^{*}_{I_{\max}}(k,s) - \\ - {C^{I_0}_{I_0}}^{-1}\Omega'^{I_r}_{I_0}\left(\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1}\Omega'^{I_r}_{I_0}\right)^{-1}\left(-\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1}C^{I_{\min}}_{I_0} + \Omega'^{I_{\min}}_{I_r}\right)\boldsymbol{u}^{*}_{I_{\min}}(k,s)\end{aligned}$$

або

$$\begin{aligned}\boldsymbol{u}^{**}_{I_0}(k,s) = {C^{I_0}_{I_0}}^{-1}\Omega'^{I_r}_{I_0}\left(\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1}\Omega'^{I_r}_{I_0}\right)^{-1}{\boldsymbol{x}^{**}_{s}}_{I_r} + \\ + {C^{I_0}_{I_0}}^{-1}\left(\Omega'^{I_r}_{I_0}\left(\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1}\Omega'^{I_r}_{I_0}\right)^{-1}\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1} - I\right)\boldsymbol{b}_{I_0} - \\ -{C^{I_0}_{I_0}}^{-1}\left(\Omega'^{I_r}_{I_0}\left(\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1}\Omega'^{I_r}_{I_0}\right)^{-1}\left(-\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1}C^{I_{\max}}_{I_0} + \Omega'^{I_{\max}}_{I_r}\right) + C^{I_{\max}}_{I_0}\right)\boldsymbol{u}^{*}_{I_{\max}}(k,s) - \\ -{C^{I_0}_{I_0}}^{-1}\left(\Omega'^{I_r}_{I_0}\left(\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1}\Omega'^{I_r}_{I_0}\right)^{-1}\left(-\Omega'^{I_0}_{I_r}{C^{I_0}_{I_0}}^{-1}C^{I_{\min}}_{I_0} + \Omega'^{I_{\min}}_{I_r}\right) + C^{I_{\min}}_{I_0}\right)\boldsymbol{u}^{*}_{I_{\min}}(k,s).\end{aligned} \tag{3.88}$$

Залишається лише перевірити виконання усіх нерівностей із (3.75).

3.4.5 Оптимізаця розрахунків

Введемо позначення

$$\Gamma(I_0) \coloneqq {C_{I_0}^{I_0}}^{-1} {\Omega'}_{I_0}^{I_r} \left({\Omega'}_{I_r}^{I_0} {C_{I_0}^{I_0}}^{-1} {\Omega'}_{I_0}^{I_r} \right)^{-1}, \tag{3.89}$$

$$\Pi(I_0) \coloneqq {C_{I_0}^{I_0}}^{-1} \left({\Omega'}_{I_0}^{I_r} \left({\Omega'}_{I_r}^{I_0} {C_{I_0}^{I_0}}^{-1} {\Omega'}_{I_0}^{I_r} \right)^{-1} {\Omega'}_{I_r}^{I_0} {C_{I_0}^{I_0}}^{-1} - I \right) \quad \text{і} \tag{3.90}$$

$$\Upsilon(I_0) \coloneqq {C_{I_0}^{I_0}}^{-1} \left({\Omega'}_{I_0}^{I_r} \left({\Omega'}_{I_r}^{I_0} {C_{I_0}^{I_0}}^{-1} {\Omega'}_{I_0}^{I_r} \right)^{-1} \left(-{\Omega'}_{I_r}^{I_0} {C_{I_0}^{I_0}}^{-1} C_{I_0} + {\Omega'}_{I_r} \right) + C_{I_0} \right). \tag{3.91}$$

Користуючись цими позначеннями, скоротимо запис (3.88), отримавши

$$\begin{aligned} \boldsymbol{u}_{I_0}^{**}(k,s) = \Gamma(I_0) {\boldsymbol{x}_s^{**}}_{I_r} + \Pi(I_0)\boldsymbol{b}_{I_0} - \\ - \Upsilon(I_0)^{I_{\max}} \boldsymbol{u}_{I_{\max}}^{*}(k,s) - \\ - \Upsilon(I_0)^{I_{\min}} \boldsymbol{u}_{I_{\min}}^{*}(k,s). \end{aligned} \tag{3.92}$$

Легко бачити, що значення матриці (3.91) залежить лише від I_0 та I_r (а останній можна обрати одним і тим же для кожного I_0). Таким чином, для обчислення усіх виразів (3.92) із однаковими I_0 потрібно обчислити значення однієї і тієї ж матриці $\Upsilon^{\overline{1..n} \setminus I_0}$. Це значно скоротить обчислення.

Підставимо (3.88) у нерівності із (3.75). Таким чином, отримуємо вирази

$$\begin{aligned} {\Omega'}_{I_{\min}} \Bigg(\Gamma(I_0) {\boldsymbol{x}_s^{**}}_{I_r} + \Pi(I_0)\boldsymbol{b}_{I_0} - \\ - \Upsilon(I_0)^{I_{\max}} \boldsymbol{u}_{I_{\max}}^{*}(k,s) - \\ - \Upsilon(I_0)^{I_{\min}} \boldsymbol{u}_{I_{\min}}^{*}(k,s) \Bigg) \leqslant {\boldsymbol{x}_s^{**}}_{I_{\min}}, \end{aligned}$$

$$\Omega'_{I_{\max}}\Bigg(\Gamma(I_0)\boldsymbol{x}^{**}_{s\ I_r}+\Pi(I_0)\boldsymbol{b}_{I_0}-$$
$$-\Upsilon(I_0)^{I_{\max}}\boldsymbol{u}^{*}_{I_{\max}}(k,s)-$$
$$-\Upsilon(I_0)^{I_{\min}}\boldsymbol{u}^{*}_{I_{\min}}(k,s)\Bigg)\geqslant\boldsymbol{x}^{**}_{s\ I_{\max}},$$

$$\Gamma(I_0)\boldsymbol{x}^{**}_{s\ I_r}+\Pi(I_0)\boldsymbol{b}_{I_0}-$$
$$-\Upsilon(I_0)^{I_{\max}}\boldsymbol{u}^{*}_{I_{\max}}(k,s)-$$
$$-\Upsilon(I_0)^{I_{\min}}\boldsymbol{u}^{*}_{I_{\min}}(k,s)\leqslant\boldsymbol{u}_{0I_0}\quad\text{і}$$

$$\Gamma(I_0)\boldsymbol{x}^{**}_{s\ I_r}+\Pi(I_0)\boldsymbol{b}_{I_0}-$$
$$-\Upsilon(I_0)^{I_{\max}}\boldsymbol{u}^{*}_{I_{\max}}(k,s)-$$
$$-\Upsilon(I_0)^{I_{\min}}\boldsymbol{u}^{*}_{I_{\min}}(k,s)\geqslant-\boldsymbol{u}_{0I_0}.$$

Перетворимо ці вирази для більшої зручності в подальшій роботі з ними, отримавши вирази (3.93), (3.94), (3.95) і (3.96).

$$\Omega'_{I_{\min}}\Bigg(-\Upsilon(I_0)^{I_{\max}}\boldsymbol{u}^{*}_{I_{\max}}(k,s)-$$
$$-\Upsilon(I_0)^{I_{\min}}\boldsymbol{u}^{*}_{I_{\min}}(k,s)\Bigg)\leqslant\boldsymbol{x}^{**}_{s\ I_{\min}}-\Omega'_{I_{\min}}\Bigg(\Gamma(I_0)\boldsymbol{x}^{**}_{s\ I_r}+$$
$$+\Pi(I_0)\boldsymbol{b}_{I_0}\Bigg),\quad(3.93)$$

$$\Omega'_{I_{\max}}\Bigg(-\Upsilon(I_0)^{I_{\max}}\boldsymbol{u}^{*}_{I_{\max}}(k,s)-$$
$$-\Upsilon(I_0)^{I_{\min}}\boldsymbol{u}^{*}_{I_{\min}}(k,s)\Bigg)\geqslant\boldsymbol{x}^{**}_{s\ I_{\max}}-\Omega'_{I_{\max}}\Bigg(\Gamma(I_0)\boldsymbol{x}^{**}_{s\ I_r}+$$
$$+\Pi(I_0)\boldsymbol{b}_{I_0}\Bigg),\quad(3.94)$$

$$- \Upsilon(I_0)^{I_{\max}} \boldsymbol{u}^*_{I_{\max}}(k,s) -$$
$$- \Upsilon(I_0)^{I_{\min}} \boldsymbol{u}^*_{I_{\min}}(k,s) \leqslant \boldsymbol{u}_{0I_0} - \left(\Gamma(I_0) \boldsymbol{x}^{**}_{s\ I_r} + \Pi(I_0) \boldsymbol{b}_{I_0} \right) \quad \text{і} \quad (3.95)$$

$$- \Upsilon(I_0)^{I_{\max}} \boldsymbol{u}^*_{I_{\max}}(k,s) -$$
$$- \Upsilon(I_0)^{I_{\min}} \boldsymbol{u}^*_{I_{\min}}(k,s) \geqslant -\boldsymbol{u}_{0I_0} - \left(\Gamma(I_0) \boldsymbol{x}^{**}_{s\ I_r} + \Pi(I_0) \boldsymbol{b}_{I_0} \right). \quad (3.96)$$

Введемо позначення

$$\boldsymbol{\eta} = \begin{pmatrix} \eta_1 \\ \eta_2 \\ \vdots \\ \eta_n \end{pmatrix}, \quad (3.97)$$

$$\eta_i = \begin{cases} -1 & i \in I_{\min} \\ 1 & i \in I_{\max} \quad \text{та} \\ 0 & i \in I_0 \end{cases} \quad (3.98)$$

$$\boldsymbol{\chi} = \boldsymbol{x}^{**}_s - \Omega' \left(\Gamma(I_0) \boldsymbol{x}^{**}_{s\ I_r} + \Pi(I_0) \boldsymbol{b}_{I_0} \right). \quad (3.99)$$

Тоді $\boldsymbol{u}^*_{\min}(k,s) = \operatorname{diag}\{\boldsymbol{u}_{0I_{\min}}\}\boldsymbol{\eta}_{I_{\min}}$ і $\boldsymbol{u}^*_{\max}(k,s) = \operatorname{diag}\{\boldsymbol{u}_{0I_{\max}}\}\boldsymbol{\eta}_{I_{\max}}$. Таким чином, (3.93) і (3.94) можна записати як

$$\operatorname{diag}\{\boldsymbol{\eta}\} \left(-\Omega' \Upsilon(I_0) \operatorname{diag}\{\boldsymbol{u}_0\} \boldsymbol{\eta} - \boldsymbol{\chi} \right) \geqslant 0,$$

а (3.95) і (3.96) - як

$$\Upsilon(I_0)\,\mathrm{diag}\{\boldsymbol{u}_0\}\boldsymbol{\eta} \leqslant \boldsymbol{u}_{0I_0} - \left(\Gamma(I_0)\boldsymbol{x}^{**}_{s\,I_r} + \Pi(I_0)\boldsymbol{b}_{I_0}\right) \quad \text{і}$$

$$\Upsilon(I_0)\,\mathrm{diag}\{\boldsymbol{u}_0\}\boldsymbol{\eta} \geqslant -\boldsymbol{u}_{0I_0} - \left(\Gamma(I_0)\boldsymbol{x}^{**}_{s\,I_r} + \Pi(I_0)\boldsymbol{b}_{I_0}\right)$$

відповідно.

Це дозволяє не виконувати прямий перебір варіантів, а натомість застосувати алгоритми пошуку розв'язків для дискретних задач задоволення обмежень.

3.5 Висновки

У даному розділі було наведено розроблені математичні методи розрахунку керування на скінченному горизонті, досліджено їх особливості, запропоновано і обгрунтовано схеми їх застосування до власне керування системою.

Отримані методи мають різні характеристики якості отриманого результату (швидкості стабілізації) і необхідних обчислювальних ресурсів. Найшвидше обчислення виконуються за допомогою варіаційного методу, але згенеровані ним проміжні цілі зазвичай знаходяться далі від $\mathbf{0}$, аніж отримані за допомогою інших методів.

Метод повного перебору був спробою позбавитись недоліків варіаційного методу, проте він не здатен згенерувати керування для випадків, коли цільова точка не досяжна за обраний горизонт.

Логічним розвитком цього методу став метод найкращої цілі. Він дозволяє з певною точністю згенерувати послідовність керувань, яка приведе систему до точки, яка знаходиться найбільш близько до цільової серед усіх досяжних. Цей метод забезпечує найкращий баланс між швидкістю виконання обчислень і якістю отриманого результату та ефективністю використання

наявного ресурсу керування. Недоліком цього методу є те, що у рідкісних випадках для синтезу керування необхідно виконати суттєво більше обчислень, аніж зазвичай. Для подолання цієї проблеми необхідно або розробити нову схему вибору обходження розбиттів, або спробувати розв'язувати поставлену у цьому методі задачу варіаційного числення у інший спосіб.

Також недоліком методу найкращої цілі, хоча і незначним, є похибка, внесена використанням регуляризуючого компонента у розв'язуємій задачі варіаційного числення. Спробою усунути цю похибку був метод послідовних оптимумів. Цей метод дозволяє отримати розв'язок цієї задачі в аналітичному вигляді, проте як виявилось, він потребує колосальних обчислювальних ресурсів і складний у програмній реалізації.

Таким чином найбільш перспективними методами виявились варіаційний і метод найкращої цілі. Варіаційний метод найкраще підходить для керування технічними системами, де необхідно синтезувати керування за малий проміжок часу. Метод найкращої цілі у його нинішньому вигляді краще підходить для систем з відносно великим інтервалом реального часу між ітераціями, але натомість більш ефективно використовує наявний ресурс керування.

4 ПРАКТИЧНЕ ЗАСТОСУВАННЯ І РЕЗУЛЬТАТИ МОДЕЛЮВАННЯ

4.1 Режими застосування

Як уже було сказано, аби врахувати вплив шумів, необхідно після кожної ітерації заново розраховувати керування. Інакше, по досягненню моменту часу $k + s$ відхилення від очікуваної траєкторії накопичиться, і у випадку нестійкої чи напівнестійкої системи ($\|A\| \geqslant 1$) може досягати суттєвих величин.

Таким чином постає питання, на який горизонт виконувати синтез керування в наступний раз і яку точку $\boldsymbol{x}^*$ обирати за цільову. Найбільш очевидним режимом застосування алгоритмів синтезу керування на фіксованому горизонті є наступний:

1) у момент часу k обчислюємо керування для моментів часу від k до $k + s - 1$ і застосовуємо перше із обчисленої послідовності керувань. Стан, до якого би прийшла система при застосуванні обчисленої послідовності за відсутності збурень позначаємо як $\boldsymbol{x}^{\#}$;
2) у моменти часу $k+i$, $i \in \overline{1..s-1}$ обчислюємо таке керування для горизонту $k + i$ - $k + s - 1$, щоби відхилення очікуваного стану системи в момент $k+s$ від $\boldsymbol{x}^{\#}$ було мінімальним. Застосовуємо перше із згенерованої послідовності керувань. Якщо i дорівнювало $s-1$, тоді присвоюємо k значення $k + s$ і переходимо до виконання першого пункту.

Таким чином ми намагаємось досягнути обрану в момент k проміжну мету $\boldsymbol{x}^{\#}$. Назвемо цю схему застосування строгою з досягненням поставленої мети.

Проте, це не найбільш раціональна стратегія. У всіх теоремах про збіжність на дальній дистанції, наведених у пунктах 2.4.4 і 3.1.1, спільним є те, що норма наступної проміжної цілі $\|\boldsymbol{x}^{\#}\|$ має бути не більше за певне обмеження. Тому нам не обов'язково зберігати ту саму проміжну ціль - достатньо мінімізувати норму стану в момент $k + s$ $\|\boldsymbol{x}_{k+s}\|$. Таким чином, у пункті 2) ми мінімізуємо не відстань очікуваного стану системи від $\boldsymbol{x}^{\#}$, а його відстань

від $\mathbf{0}$, тобто норму. Такий підхід є більш раціональним, бо останні величини шумів можуть грати нам на руку, а після їх здійснення $\boldsymbol{x}^{\#}$ може виявитись не найближчим значенням до $\mathbf{0}$, якщо взагалі досяжним. Цю схему називатимемо строгою з досягненням мінімуму.

Існує також експериментальна схема, згідно з якою керування також переобчислюється після кожної ітерації системи, але довжина горизонту при кожному наступному обчисленні не змінюється. Назвемо її асимптотичною. Ця схема не обґрунтовується доведеними у цій роботі теоремами про збіжність на дальній дистанції, але згідно обчислювальним експериментам також цю збіжність забезпечує.

Якщо точність вимірювання стану буде співрозмірною з розмірами випадкових збурень, є сенс продовжувати використовувати раніше обчислену послідовність керувань, якщо відхилення від прогнозованої траєкторії не перевищує розміри похибки.

4.2 Випробування роботи алгоритмів

Випробування проводилось на модельній когнітивній карті, що описується матрицями

$$A = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ -0,02533229 & 0,08386109 & -0,44968984 & 0,70611216 & -1,305778 & 0,6368 & 0,59 \end{pmatrix},$$

$$B = \begin{pmatrix} -0{,}88 & 0{,}73 & -0{,}84 \\ -0{,}39 & 0{,}44 & -0{,}56 \\ 0{,}2 & 0{,}69 & -0{,}16 \\ -0{,}48 & 0{,}24 & -0{,}8 \\ -0{,}38 & 0{,}78 & -0{,}8 \\ 0 & -0{,}46 & 0{,}09 \\ 0{,}25 & -0{,}87 & 0{,}74 \end{pmatrix},$$

а також із рівномірно розподіленими збуреннями з інтервалом $[-0,01; 0,01]$. Тут матриця A - транспонована матриця Фробеніуса, сформована таким чином, щоб її власні числа були $-1,27; 0,77 \pm 0,53i; 0,08 \pm 0,48i; 0,08 \pm 0,3i$. Таким чином, матриця A є нестійкою. Синтез керувань виконується для горизонтів довжиною до $s = 6$ ітерацій.

На рисунках 4.1, 4.2, 4.3, 4.4, 4.5 та 4.6 зображено результати моделювання для наведених у підрозділі 4.1 схем керування із застосуванням методу найкращої цілі. Також на рисунках 4.8 та 4.7 зображено результати моделювання при застосуванні усієї послідовності згенерованих керувань як є, тобто без зворотнього зв'язку.

На цих рисунках дійсна траєкторія зміни стану системи показана синьою кривою, а прогнозовані значення для горизонтів різної довжини показані відтінками від чорного до світло-сірого. Чорним кольором показано прогноз на 6 кроків уперед, а найсвітлішим - на один крок.

4.3 Висновки

У цьому розділі було запропоновано схеми застосування алгоритмів синтезу керувань, які забезпечують зворотній зв'язок. Було виконано моделювання керування нестійкою системою для цих схем, а також для схеми без зворотнього зв'язку.

Як видно з результатів моделювання, найкращі результати дала асимптотична схема. Дещо гірші результати дають строга схема з досягненням

мінімуму і строга схема з досягненням поставленої мети. Найгірші результати дало застосування схеми без зворотнього зв'язку.

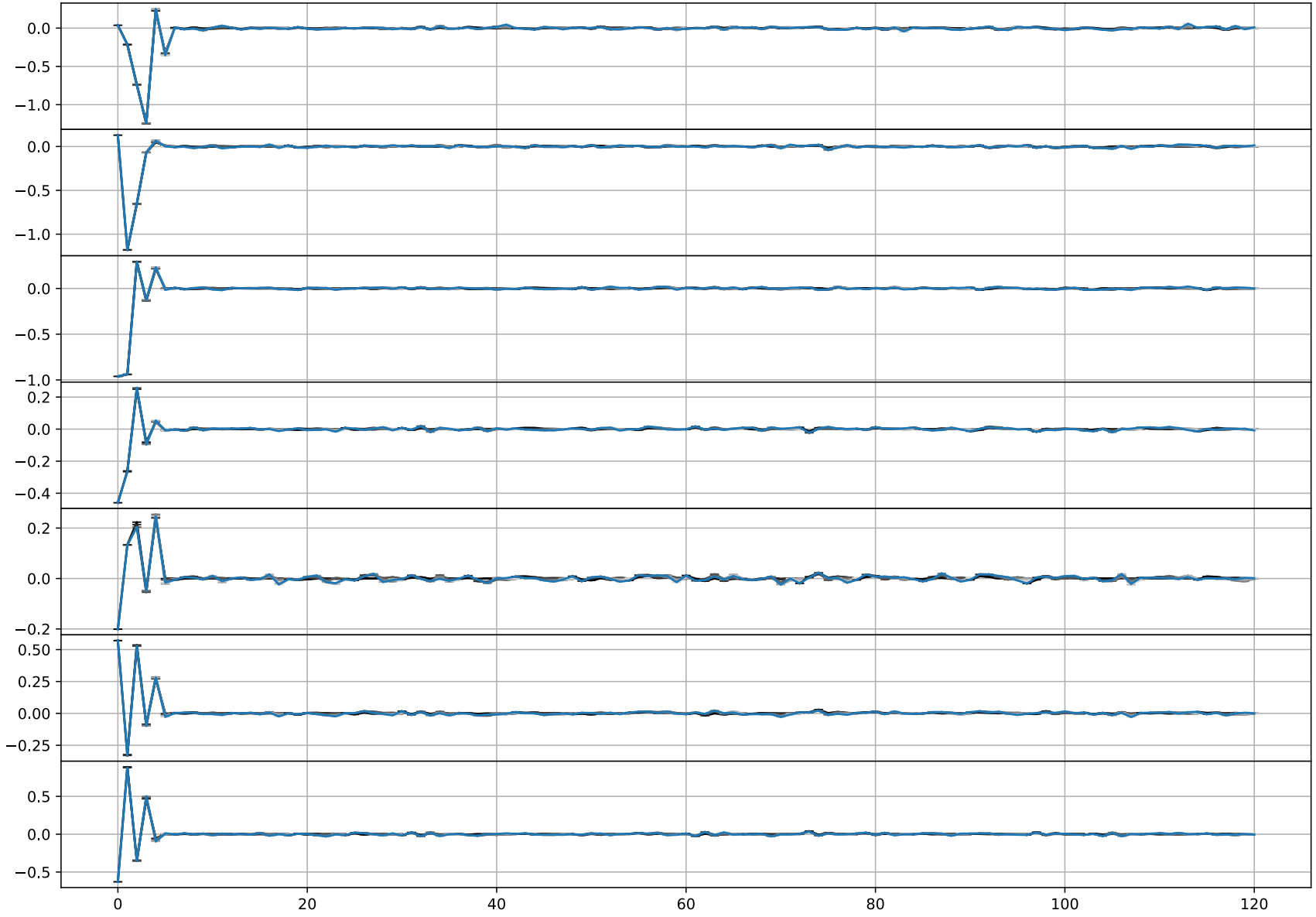

Рисунок 4.1 — Приклад керування когнітивною картою за строгою схемою з досягненням поставленої мети, значення $\Delta \boldsymbol{x}$.

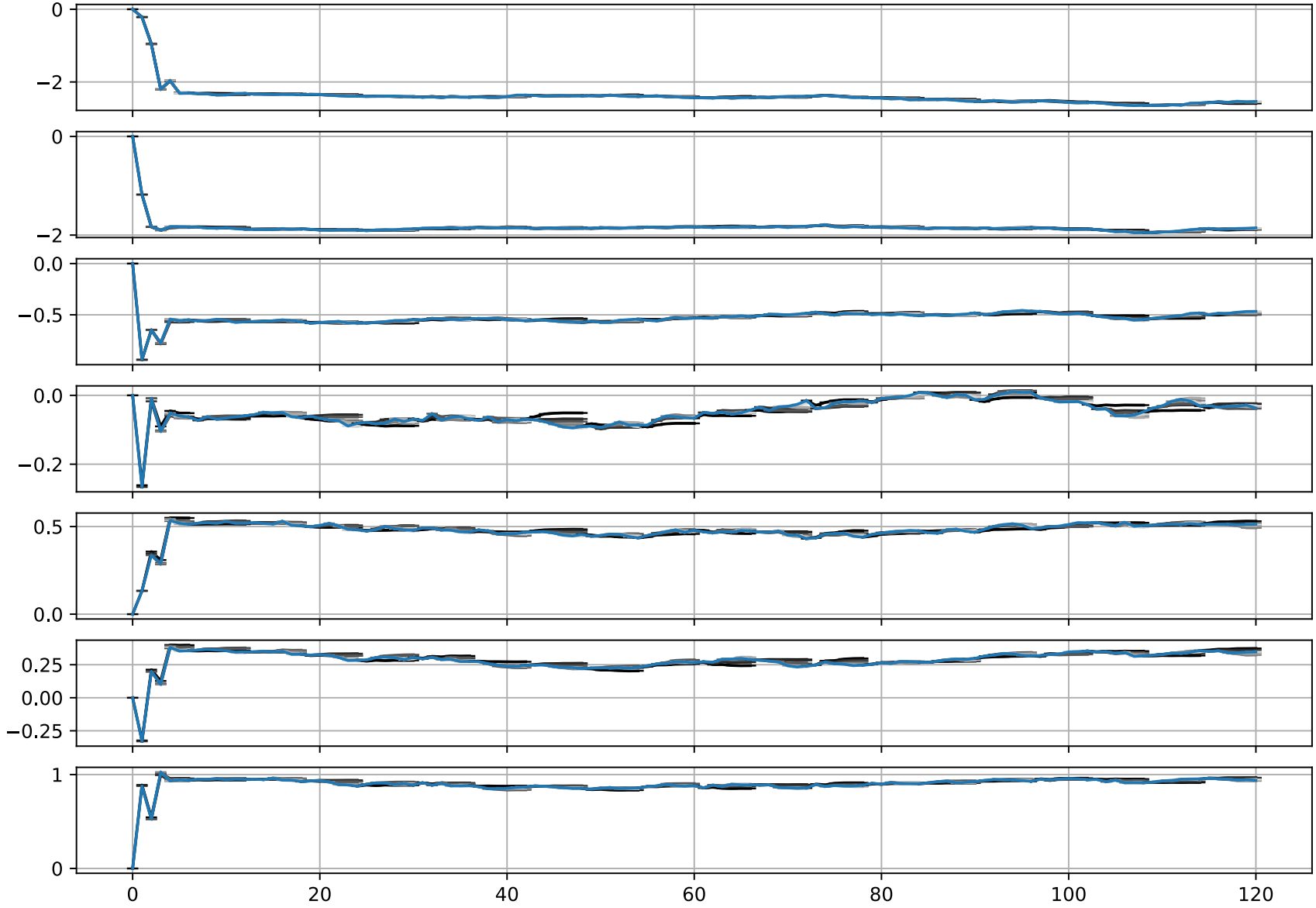

Рисунок 4.2 — Приклад керування когнітивною картою за строгою схемою з досягненням поставленої мети, значення $\boldsymbol{x}$.

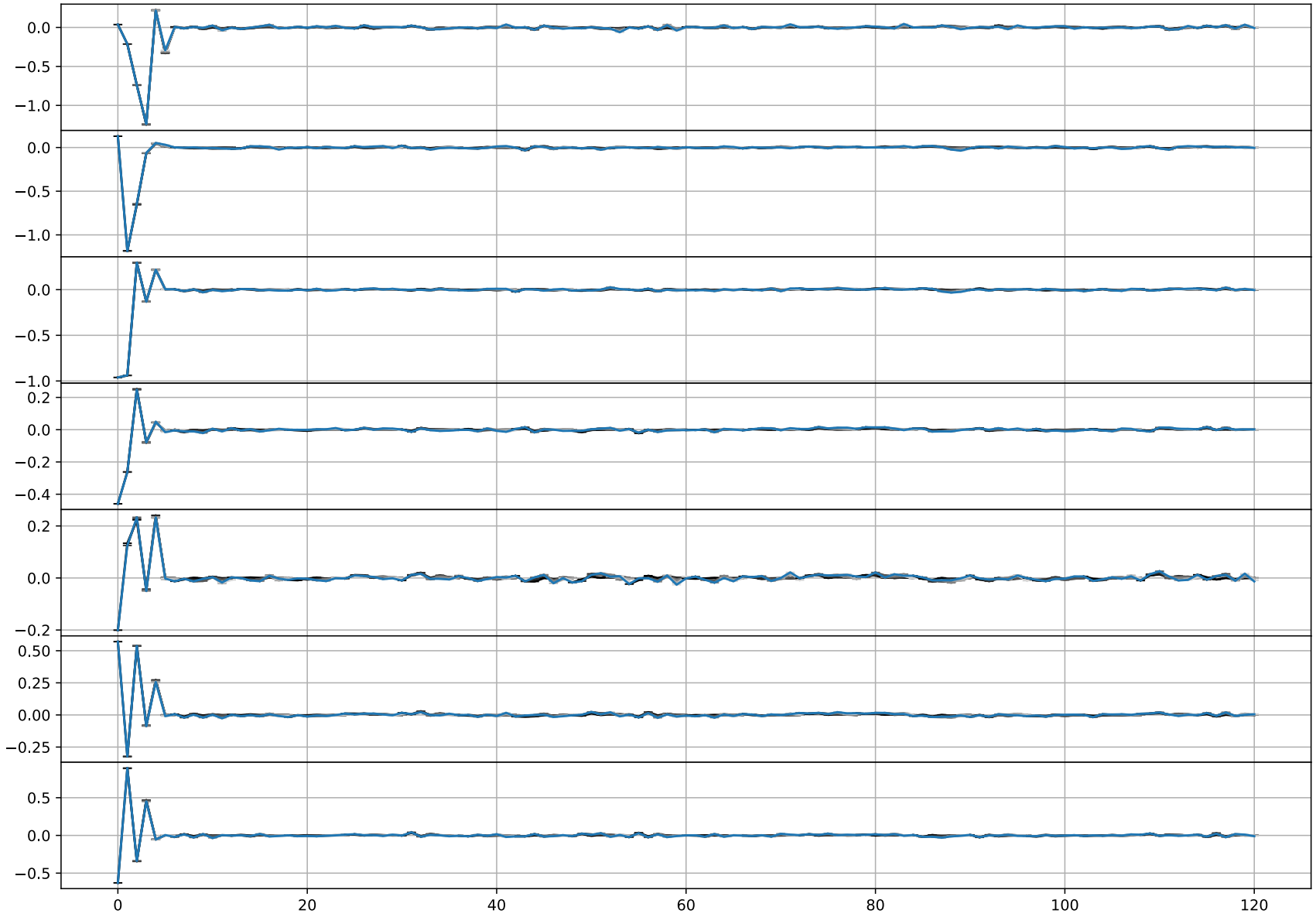

Рисунок 4.3 — Приклад керування когнітивною картою за строгою схемою з досягненням мінімуму, значення $\Delta \boldsymbol{x}$.

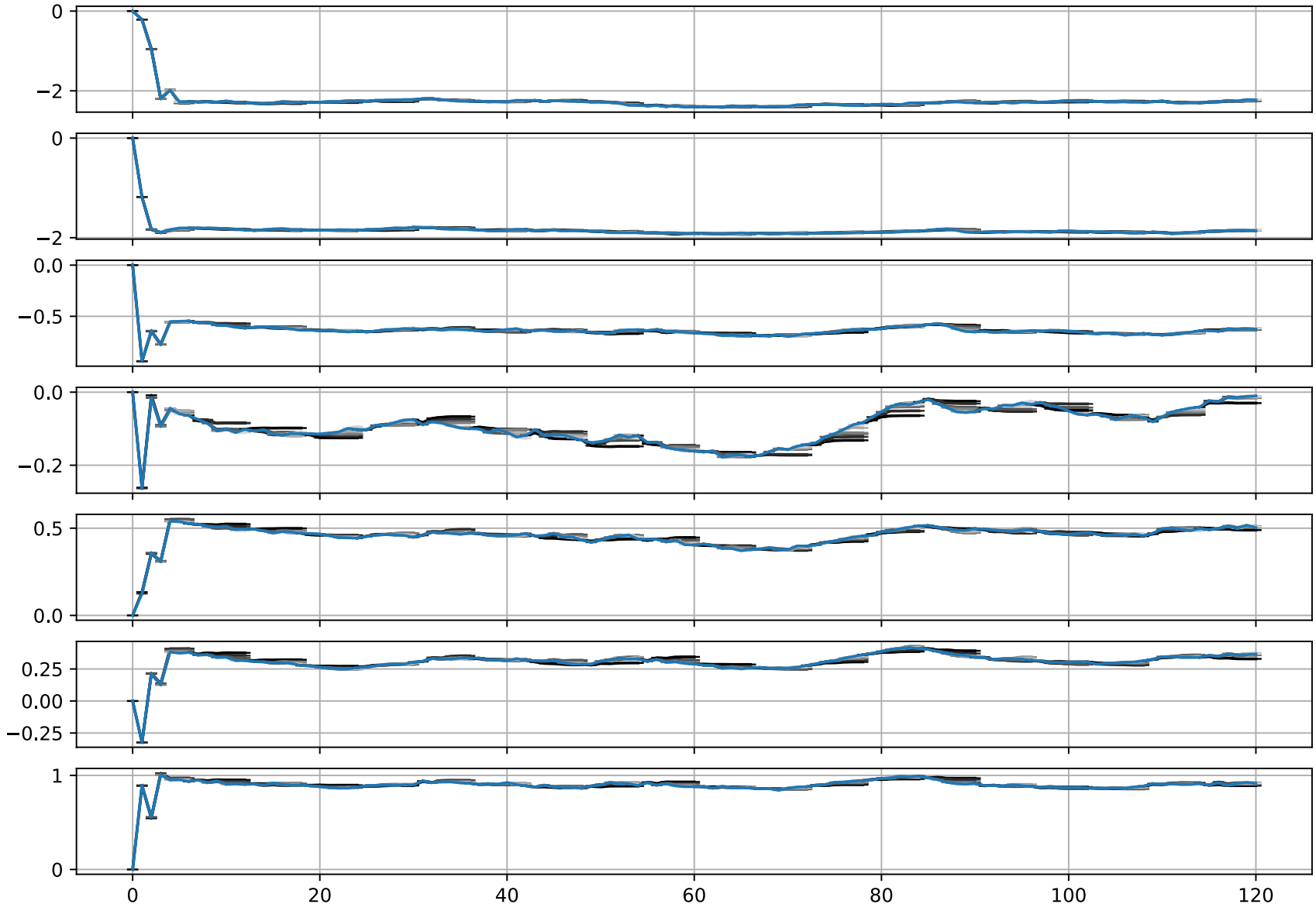

Рисунок 4.4 — Приклад керування когнітивною картою за строгою схемою з досягненням мінімуму, значення $\boldsymbol{x}$.

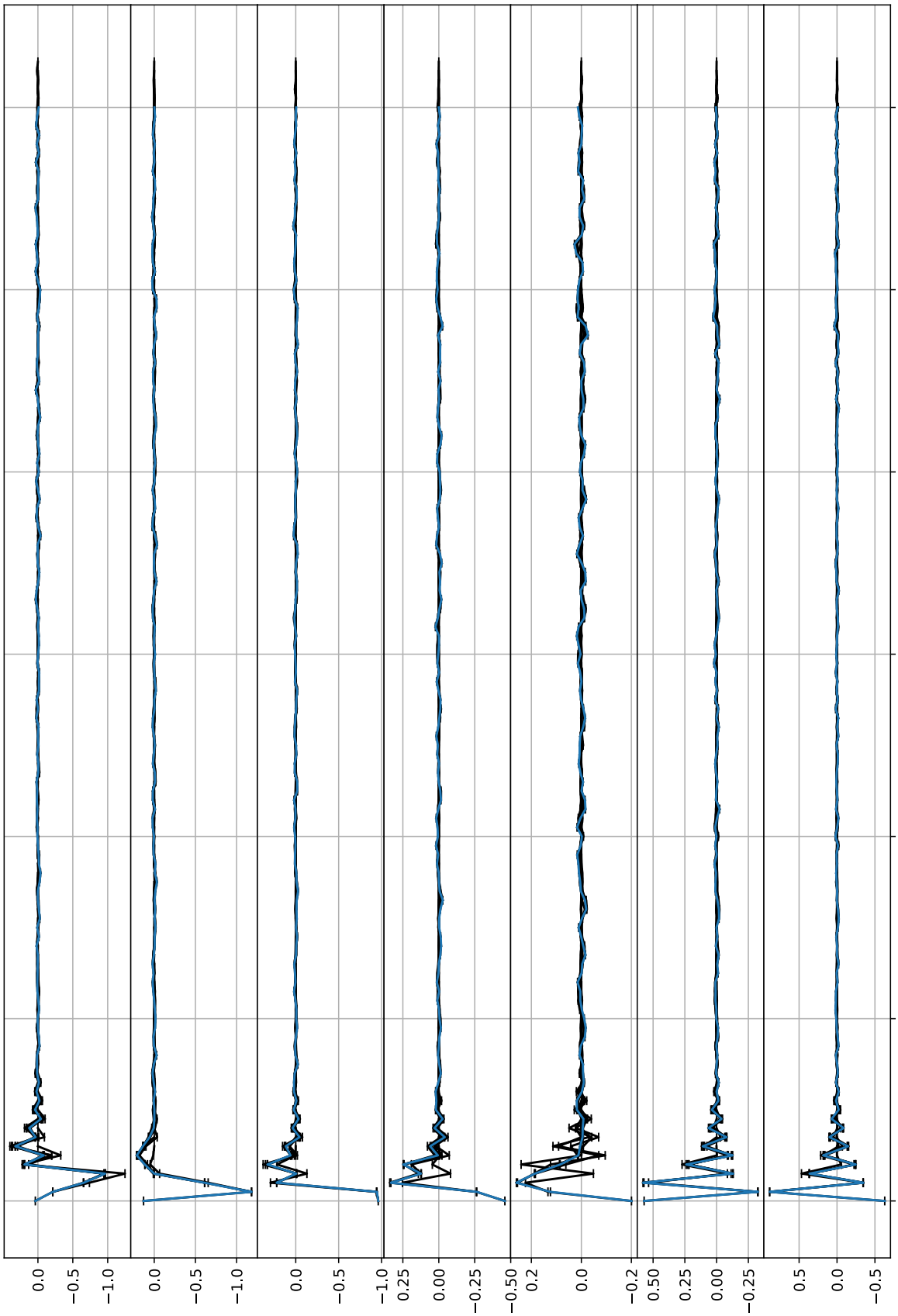

Рисунок 4.5 — Приклад керування когнітивною картою за асимптотичною схемою, значення $\Delta \boldsymbol{x}$.

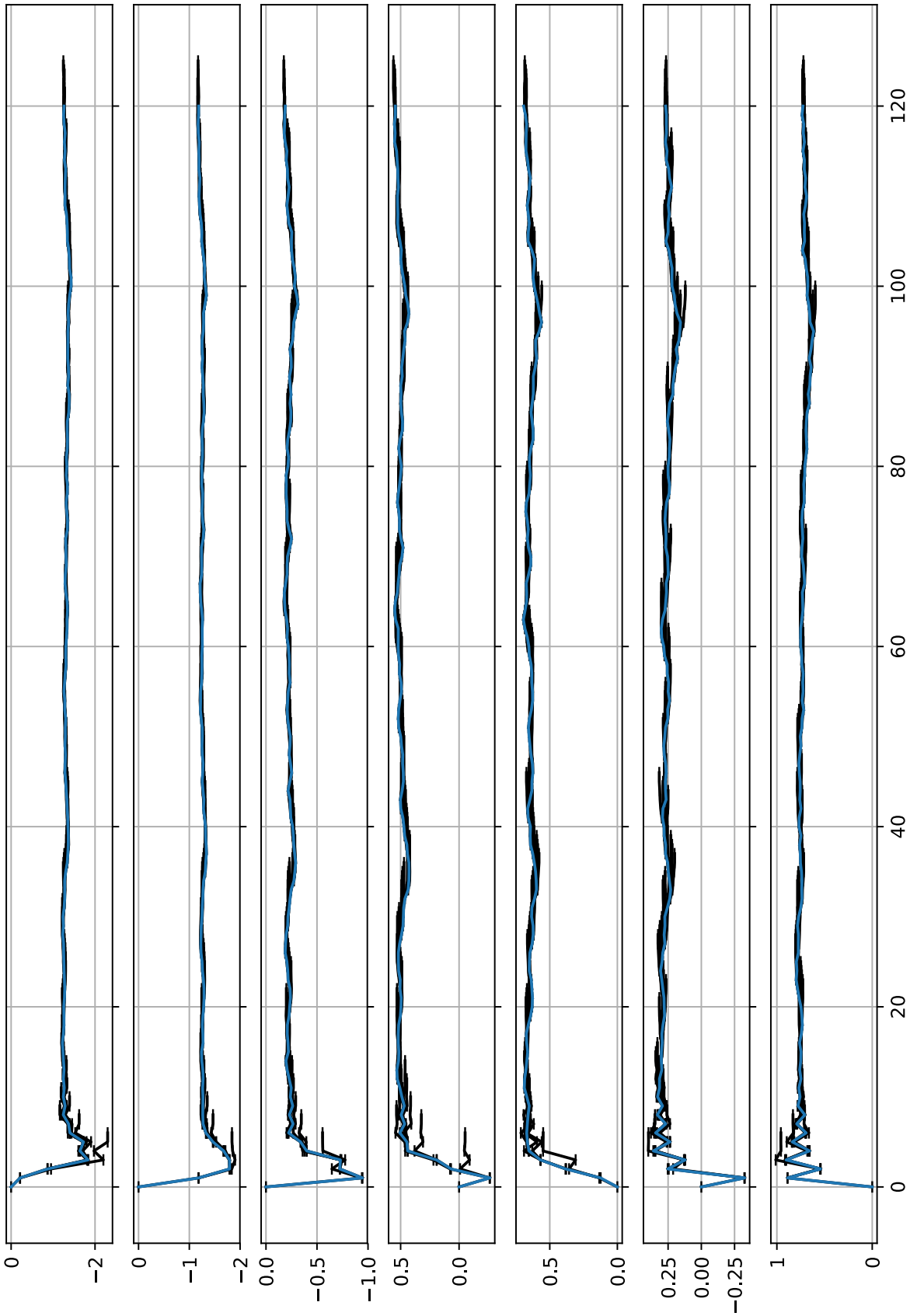

Рисунок 4.6 — Приклад керування когнітивною картою за асимптотичною схемою, значення $\boldsymbol{x}$.

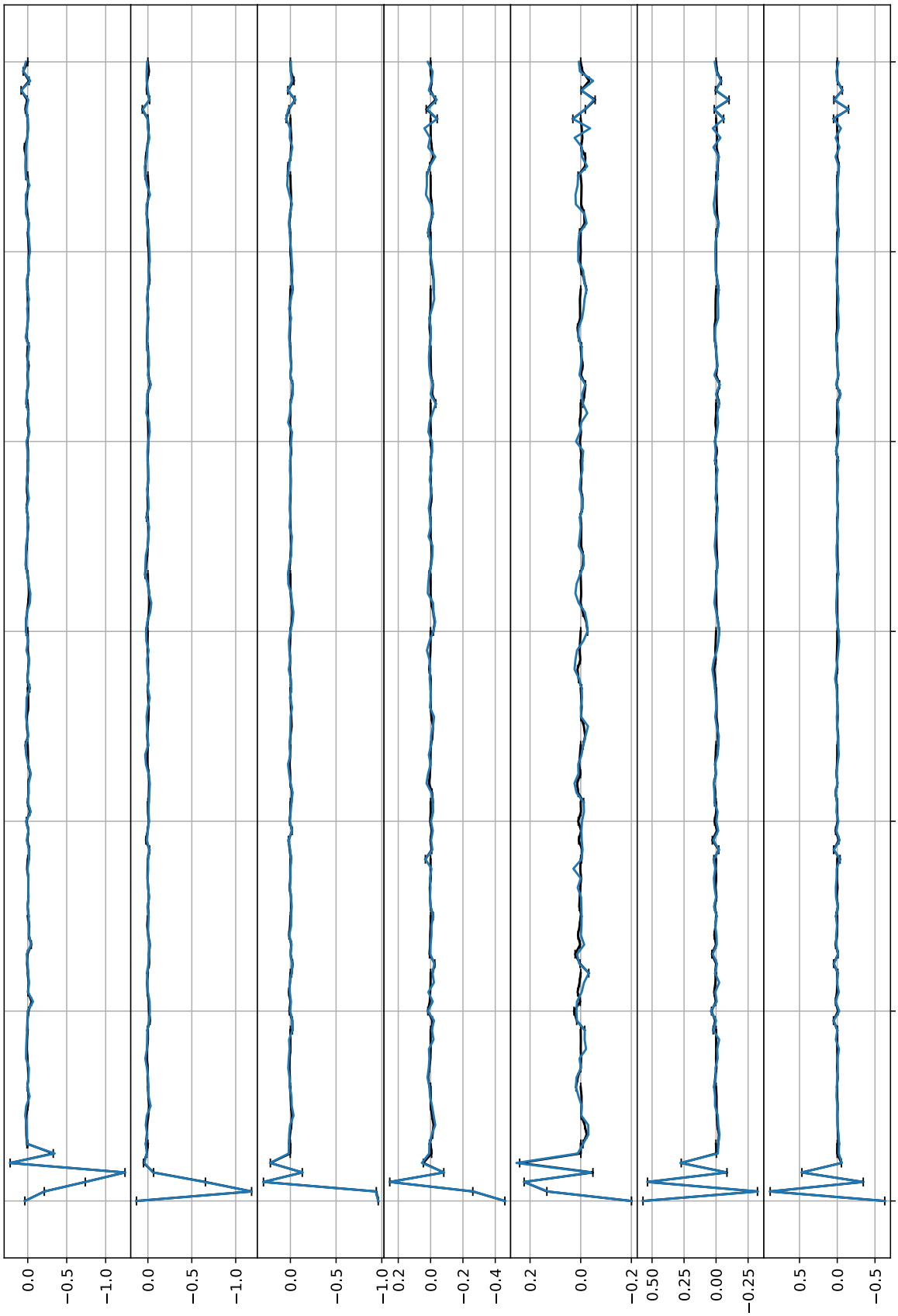

Рисунок 4.7 — Приклад керування когнітивною картою без зворотнього зв'язку, значення $\Delta \boldsymbol{x}$.

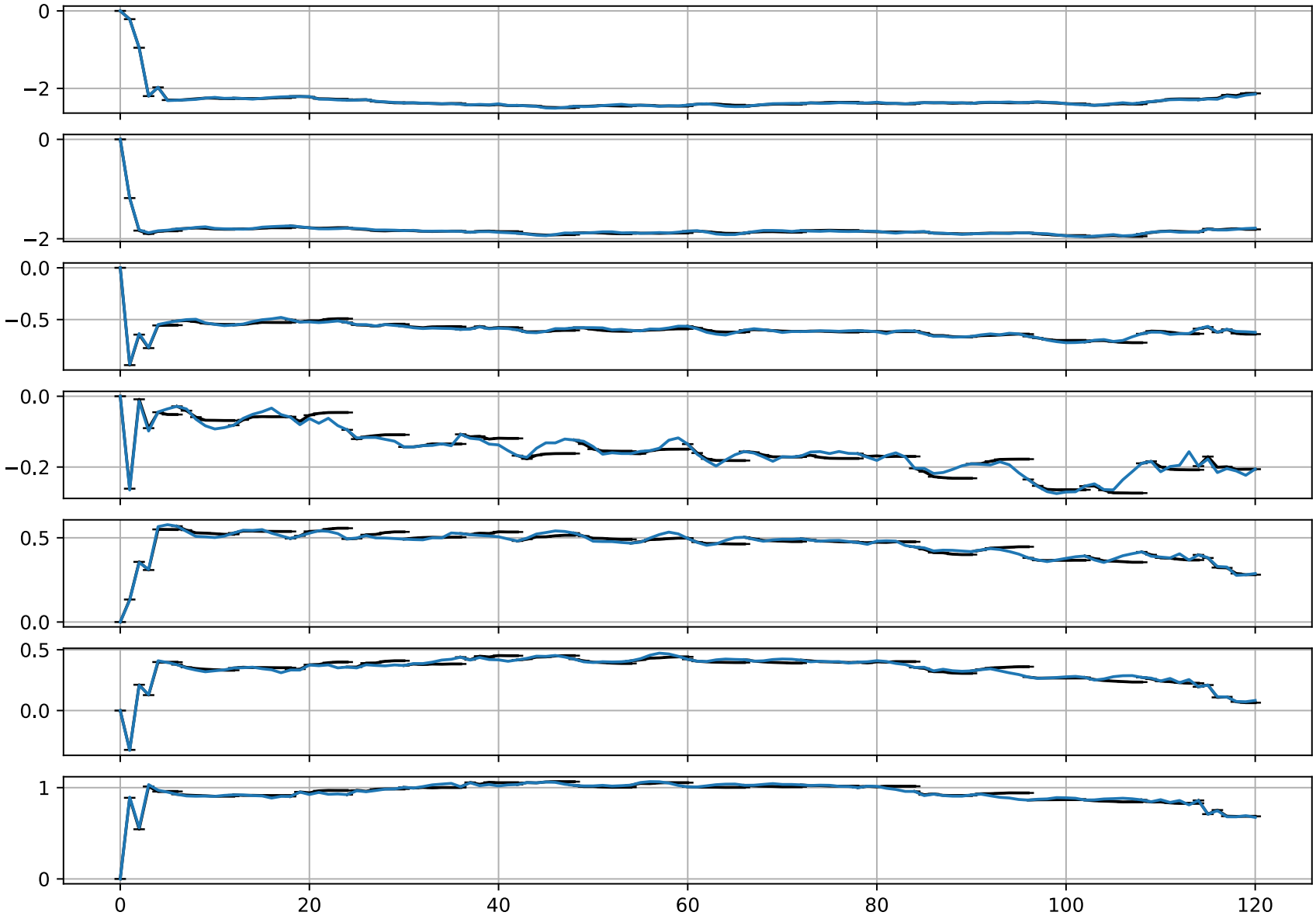

Рисунок 4.8 — Приклад керування когнітивною картою без зворотнього зв'язку, значення $\boldsymbol{x}$.

ВИСНОВКИ

У даній роботі було сформульовано нові підходи до керування лінійними системами та когнітивними картами з урахуванням обмежень на величини керувань, розроблено відповідні обчислювальні методи і схеми їх застосування, проведено дослідження їх роботи, що продемонстрували їх працездатність.

Було досягнуто можливості керування у тому числі і нестійкими системами в умовах наявності зовнішніх збурень. Те, що, розроблені методи не виходять із припущень про будь які статистичні властивості збурень, окрім їх обмеженності, є однією з їх ключових переваг і особливостей.

Запропоновані підходи відкривають широку область для досліджень у даній галузі.

ПЕРЕЛІК ДЖЕРЕЛ ПОСИЛАННЯ

1. Карманов В. Г. Математическое программирование. Москва: Изд-во физ.-мат. литературы, 2004. 264 с.
2. Gubarev V.F., Mishchenko M.D., Snizhko B.M. (2019) Model Predictive Control for Discrete MIMO Linear Systems. *Advanced Control Techniques in Complex Engineering Systems: Theory and Applications. Studies in Systems, Decision and Control.* / Ed. Kondratenko Y., Chikrii A., Gubarev V., Kacprzyk J. Cham, Springer, 2019. Vol 203. P. 63-81. DOI: https://doi.org/10.1007/978-3-030-21927-7_4
3. L. Vandenberghe The CVXOPT linear and quadratic cone program solvers, March 20, 2010. URL: http://www.ee.ucla.edu/ vandenbe/publications/coneprog.pdf
4. C.E. Garcia, D.M. Prett, M. Morari Model predictive control: Theory and Practice – a survey. *Automatica.* 1989. Vol 25. P. 335-347.
5. J.B. Rawlings, K.R. Muske The stability of constrained receding horizon control. *IEEE Transactions on Automatic Control.* Institute of Electrical and Electronics Engineers, Oct 1993. Vol 38, Issue 10. P. 1512-1516.
6. D.Q. Mayne Optimization in model based control. *IFAC Symposium*, Helsingor, Denmark, June 1995. P. 229-242.
7. T.J.J. van den Boom Model based predictive control: Status and perspective. *CESA'96 IMACS Multiconference*: Symposium on Control, Optimization and Supervision. Lille, 1996, P. 1-12.
8. J. Richalet et. al. Model predictive heuristic control: Application to industrial processes. *Automatica.* 1978. Vol 14, No. 2. P. 413-428.
9. J.B. Rawlings, D.Q. Mayne: Model Predictive Control: Theory and Design. Madison WI: Nob Hill Publishing, 2009.

10. S.S. Keerthi, E.G. Gilbert Optimal infinite-horizon feedback laws for a general class of constrained discrete-time systems: Stability and moving-horizon approximations. *Journal of Optimization Theory and Applications.* 1988. Vol 57, Issue 2, P. 265-293.

11. В.М. Кунцевич Управление в условиях неопределенности: гарантированные результаты в задачах управления и идентификации. Київ: Наукова думка, 2006. 264 с.

12. V.F. Gubarev et. al. State estimation for Systems Subjected to Bounded Uncertainty using Moving Horizon Approach. *Preprints of the 15th IFAC Symposium on System Identification.* Saint-Malo, France, 2009. P. 910-915

13. V.M. Kuntsevich et. al. Control Systems: Theory and Applications. Series in Automation, Control and Robotics. River Publishers, 2018. 329 P.

14. V.M. Kuntsevich, A.B. Kurzhanski Attainability Domains for Linear and Some Classes of Nonlinear Discrete Systems and Their Control. *Journal of Automation and Information Sciences.* 2010. Vol 42, Issue 1. P. 1-18. DOI: 10.1615/JAutomatInfScien.v42.i1

ДОДАТОК А: КОД ПРОГРАМНОЇ РЕАЛІЗАЦІЇ ДЕЯКИХ ЗАПРОПОНОВАНИХ АЛГОРИТМІВ

A.1 solvers/abstract_solver.py

```
from abc import ABCMeta, abstractmethod

class Solver(metaclass=ABCMeta):

    intended_model = None

    def __init__(self, problem, *args, **kwargs):

        self.problem = problem

    @abstractmethod
    def solve(*args, **kwargs):
        pass
    @abstractmethod
    def display(self):
        pass
```

A.2 solvers/optimalaim.py

```
from .abstract_solver import Solver
from ..onearg_cache import Cache
from scipy.linalg import inv, cho_factor, cho_solve
import numpy as np
from numpy.linalg import LinAlgError
import numpy.random as rndm

from functools import lru_cache, partial
from itertools import chain, zip_longest, islice

from warnings import warn

from .manager import Manager
from .manager.states import MultisetStateFactory
from .manager.pathchoosers import FirsttimePathChooser
from .manager.pathchoosers import RandomPathChooser
from .manager.pathchoosers import VotePathChooser
from .manager.pathchoosers import StatisticPathChooser
from .manager.pathchoosers import ExitPathChooser
```

```
from .manager.pathchoosers import ArrayIterPathChooser
from .manager.pathchoosers import AbstractSeqPathChooser

from .manager.history import SequenceHistory, ArrayHistory, MetadataHistory
from .manager.history import StatisticSeqHistory
from .manager.states  import State

from ..controllers.computing_log import PrefixProxyComputingLog

class SimultaneousPathChooser(AbstractSeqPathChooser):
    ''' A path chooser for Manager class. '''
    def __init__(self, u_max,
                       seq_hist_name        = 'seq',
                       membership_hist_name = 'array',
                       metadata_hist_name   = 'meta' ):

        super().__init__('simultaneous',
                         loopstate_meta = 'simultaneous',
                         seq_hist_name = seq_hist_name,
                         metadata_hist_name = metadata_hist_name,
                        )

        self.u_max = u_max
        self.membership_hist_name = membership_hist_name

        self.repeated_visits = False

    def __call__(self, histories, state_factory, u0=None, w=None,
                       logger = None, **kwargs):

        if u0 is None or w is None:
            return None, None

        ##################
        u_max = self.u_max
        seq_hist_name = self.seq_hist_name
        membership_hist_name = self.membership_hist_name
        metadata_hist_name = self.metadata_hist_name
        ##################

        prev_state, prev_comment = histories[seq_hist_name].last_decision

        if prev_state is None:
            return None, None

        I0, Imin, Imax = prev_state.to_sets()

        Iminmax     = Imin | Imax

        bool_eye = np.eye(len(I0), dtype=bool  )
        w_to_I  = bool_eye[:, Iminmax]
        u0_to_I = bool_eye[:, I0]

        Iminto0 = ( w_to_I @ (w <= 0) ) & Imin
        Imaxto0 = ( w_to_I @ (w >= 0) ) & Imax

        I0tomin = ( u0_to_I @ (u0 <= - u_max) ) & I0
```

```
            I0tomax = ( u0_to_I @ (u0 >=   u_max) ) & I0

            Imin = (Imin & ~Iminto0) | I0tomin
            Imax = (Imax & ~Imaxto0) | I0tomax
            I0   = (I0 & ~I0tomin & ~I0tomax) | Iminto0 | Imaxto0

            st = state_factory.to_state((I0, Imin, Imax))
            if st in histories[membership_hist_name]:

                histories[metadata_hist_name]\
                         .last_metadata[self.loopstate_meta] = \
                                                (st, self.history_comment)

                histories[membership_hist_name].inc_marker()

                if st != prev_state:
                    self.repeated_visits = True

                if logger is not None:
                    logger.inc_counter('loops')

                return None, None
            else:
                return st, self.history_comment

        def logger_init_hook(self, logger=None):
            if logger is not None:
                logger.split_counter('loops')

    class OptimalAimSolver(Solver):
        SigmaQ_cache    = Cache()
        SigmaQinv_cache = Cache()

        def __init__(self, problem, **kwargs):
            super().__init__(problem, **kwargs)

            try:
                self.SigmaQ = self.SigmaQ_cache[self.problem.Sigma]
            except KeyError:
                self.SigmaQ = self.problem.Sigma.T @ self.problem.Sigma
                self.SigmaQ_cache[self.problem.Sigma] = self.SigmaQ

            self.SigmaQ = np.asarray(self.SigmaQ)

            # Optimization of this multiplication
            # is already implemented in numpy.
            self.x_dstar = self.problem.Sigma.T @ self.problem.x_dstar
            self.x_dstar = np.asarray(self.x_dstar).ravel()

            self.I_history=dict()

            self.pivoted = False
            self._multicluster = False

        @property
        def multicluster(self):
            return self._multicluster or self.pivoted
```

```
@multicluster.setter
def multicluster(self, val):
    self._multicluster = val

def solve(self, *args, **kwargs):
    u, st = self.solve_full(*args, **kwargs)
    return u

@staticmethod
def search_mode_to_manager(state_len, u_max,
                           search_mode='pivot', I_hint=None,
                           histories=None,
                           logger=None):
    st_factory = MultisetStateFactory(state_len, 3)

    if I_hint is not None:
        I_hint = st_factory.to_state(I_hint)

    ft  = FirsttimePathChooser(start_state=I_hint)
    spc = SimultaneousPathChooser( u_max=u_max )

    if search_mode == 'oneshot':
        path_choosers = (ft,
                         spc,
                         ExitPathChooser())
        hist_classes = {
            'seq'  : SequenceHistory,
            'array': ArrayHistory,
            'meta' : MetadataHistory
        }
    elif search_mode == 'pivot':
        path_choosers = (ft,
                         spc,
                         ExitPathChooser(),
                         RandomPathChooser(),
                         ArrayIterPathChooser())
        hist_classes = {
            'seq'  : SequenceHistory,
            'array': ArrayHistory,
            'meta' : MetadataHistory
        }
    elif search_mode == 'vote':
        path_choosers = (ft,
                         spc,
                         exit_pathchooser,
                         VotePathChooser(),
                         RandomPathChooser(),
                         ArrayIterPathChooser())
        hist_classes = {
            'stat'  : StatisticSeqHistory,
            'array': ArrayHistory
        }
    elif search_mode == 'stat':

        ft  = FirsttimePathChooser(start_state=I_hint,
                                   hist_to_ask_name='stat')
        spc = SimultaneousPathChooser(
```

```
                    u_max=self.problem.u_max,
                    seq_hist_name='stat',
                    metadata_hist_name='stat'  )
        exit_pc = ExitPathChooser(seq_hist_name='stat',
                                  Wmetadata_hist_name='stat')

        path_choosers = (ft,
                         spc,
                         exit_pc,
                         StatisticPathChooser(),
                         RandomPathChooser(),
                         ArrayIterPathChooser())
        hist_classes = {
            'seq'  : SequenceHistory,
            'array': ArrayHistory,
            'meta' : MetadataHistory
        }
    elif search_mode == 'map':
        path_choosers = (ft,
                         spc,
                         VotePathChooser(),
                         ArrayIterPathChooser())
        hist_classes = {
            'seq'  : SequenceHistory,
            'array': ArrayHistory,
            'meta' : MetadataHistory
        }

    return Manager(st_factory, path_choosers, hist_classes,
                   histories=histories, logger=logger)

def solve_full(self, alpha=0.001, I_hint=None, u_hint=None,
               lru_maxsize=128, search_mode='pivot', manager=None,
               histories=None, logger=None):

    if I_hint is None and u_hint is not None:
        I_hint = self.state_factory\
                     .u_to_tristate(u_hint, self.problem.u_max)

    if manager is None:
        manager = self.search_mode_to_manager(self.problem.Sigma  \
                                                  .shape[1],
                                              self.problem.u_max,
                                              search_mode, I_hint,
                                              histories,
                                              logger=logger)

    try:
        SigmaQinv, SigmaQinv2 = self.SigmaQinv_cache[self.SigmaQ, alpha]
    except KeyError:
        SigmaQbiased =   np.diagflat((alpha,)*self.SigmaQ.shape[0])\
                       + self.SigmaQ
        SigmaQinv = inv(SigmaQbiased)

        @lru_cache(lru_maxsize)
        def SigmaQinv2(Iminmax):
            return cho_factor(SigmaQinv[Iminmax, :][:, Iminmax])
```

```
    self.SigmaQinv_cache[self.SigmaQ, alpha] = SigmaQinv, SigmaQinv2

self.I_history[alpha] = manager

#SigmaQinv2 = lambda Iminmax: SigmaQinv2(tuple(Iminmax))

# Maybe move u0_init into cache too?
# Then this cache must be instance—wide, not class—wide,
# because its value depends on changing `x_dstar`.
u0_init = SigmaQinv @ self.x_dstar


bool_eye = np.eye(self.problem\
                      .Sigma  \
                      .shape[1], dtype=bool  )

result = None, None

def compute_state(st):

    I0, Imin, Imax = st.to_sets()

    Iminmax     = Imin | Imax
    Iminmax_red = Imax[Iminmax]
    u_minmax = np.where(Iminmax_red,
                          self.problem.u_max,
                        - self.problem.u_max )

    if np.any(Iminmax):
        v = u_minmax - u0_init[Iminmax]
        w = cho_solve(SigmaQinv2(tuple(Iminmax)), v, overwrite_b=True)
    else:
        w = np.empty((0,), dtype=u0_init.dtype)
    #w = self.SigmaQ[I0[:,0],:][:,Iminmax[:,0]] @ v
    u0 =   u0_init[I0] \
         + SigmaQinv[I0,:][:,Iminmax] @ w

    # is solution?

    w_to_I  = bool_eye[:, Iminmax]
    u0_to_I = bool_eye[:, I0]

    Iminto0 = ( w_to_I @ (w <= 0) ) & Imin
    Imaxto0 = ( w_to_I @ (w >= 0) ) & Imax

    I0tomin = ( u0_to_I @ (u0 <= - self.problem.u_max) ) & I0
    I0tomax = ( u0_to_I @ (u0 >=   self.problem.u_max) ) & I0

    is_solution = not np.any(Iminto0) and \
                  not np.any(Imaxto0) and \
                  not np.any(I0tomin) and \
                  not np.any(I0tomax)

    if is_solution:
        u = np.empty_like(u0_init)
        u[I0]   = u0
```

```
            u[Imin] = - self.problem.u_max
            u[Imax] =   self.problem.u_max
        else:
            u = None

        return u0, w, u



    u0, w = None, None
    states_checked = 0

    while True:
        st, comment = manager.next(u0=u0, w=w, logger=logger)

        if st is None or st is 'exit':
            break
        u0, w, solution = compute_state(st)
        states_checked += 1

        if solution is not None:
            if result[0] is not None or \
               result[1] is not None    :
                warn(UserWarning('result obtained again', result))
            result = solution, st

            if logger is not None:
                logger.append_value('states to find solution',
                                    states_checked)


    if logger is not None:
        logger.append_value('states checked', states_checked)

    return result

def display(self):
    print('I_history:')
    for key, val in self.I_history.items():
        print('\t'+str(key)+':')
        for s in val.to_text():
            print('\t', s, sep='')

def display_stats(self):
    print('STATS:')
    for key, val in self.I_history.items():
        print('\t'+str(key)+':')
        for s in val.stat_to_text():
            print('\t', s, sep='')

def hist_to_dot(self, alpha = None):

    if alpha is not None:
        alpha_lst = (alpha,)
    else:
        alpha_lst = self.I_history.keys()
```

```
        lst = ['// ' + s for s in str(self.problem).split('\n')]
        lst.append('')

        for alpha in alpha_lst:
            lst.append('// alpha = '+str(alpha))
            lst.extend(self.I_history[alpha].to_dot())
        return lst

class OptimalAimIterSolver(OptimalAimSolver):
    def solve(self, alpha=0.001, I_hint=None,
                    lru_maxsize=128, alpha_mul=0.5,
                    logger=None, **kwargs):

        alpha_logger = None
        u = None

        J_dist = np.inf

        while True:
            if logger is not None:
                alpha_logger = PrefixProxyComputingLog(
                                    logger,
                                    'alpha={} '.format(alpha)
                                )

            try:
                next_u, next_I = self.solve_full(alpha, I_hint, lru_maxsize,
                                        logger=alpha_logger, **kwargs)
            except LinAlgError:
                break

            next_J_dist = self.problem.J_distance(u)
            if next_J_dist > J_dist:
                break

            u, I, J_dist = next_u, next_I, next_J_dist

            I_hint = I
            self.min_valid_alpha = alpha
            alpha *= alpha_mul

        if logger is not None:
            logger.append_value('min valid alpha',
                                self.min_valid_alpha)
            logger.append_value('min solution distance',
                                J_dist)
        return u

    def display(self):
        super().display()
        print('min_valid_alpha:', self.min_valid_alpha)

from .linear_problem_mpc_solver import LinearProblemMPCSolver

class OptimalAimLPSolver(LinearProblemMPCSolver, OptimalAimSolver):
    pass
```

```
class OptimalAimIterLPSolver(LinearProblemMPCSolver, OptimalAimIterSolver):
    pass
```